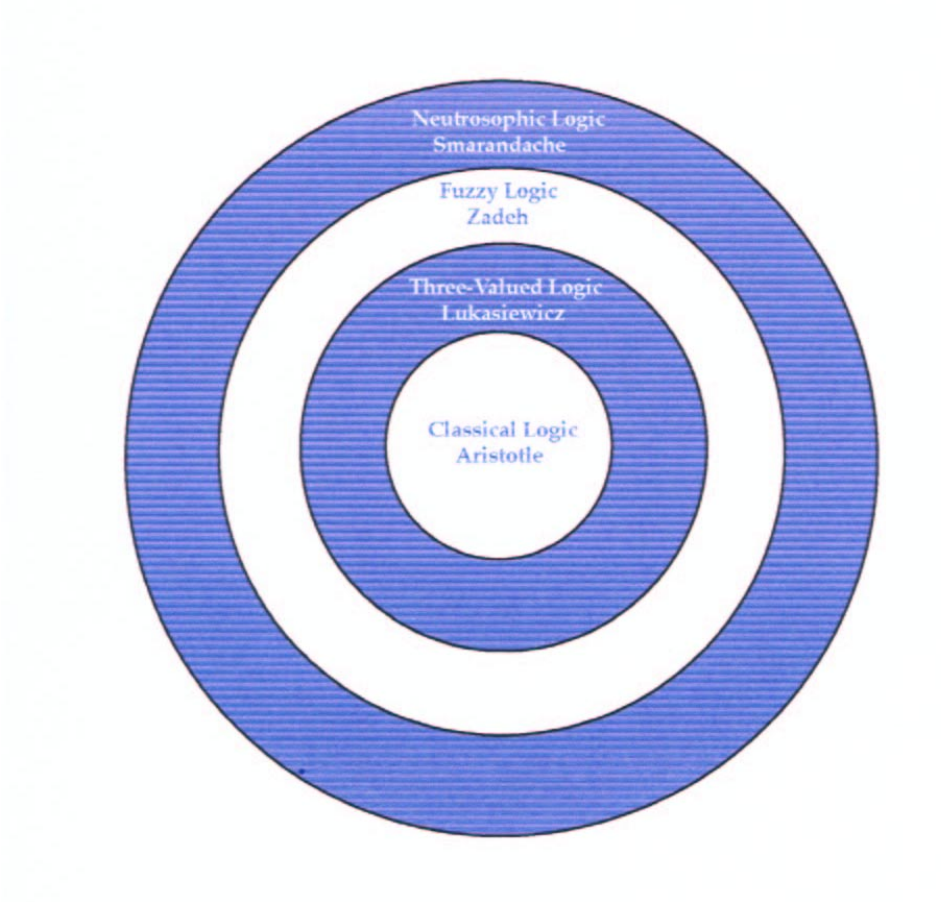
Charles Ashbacher

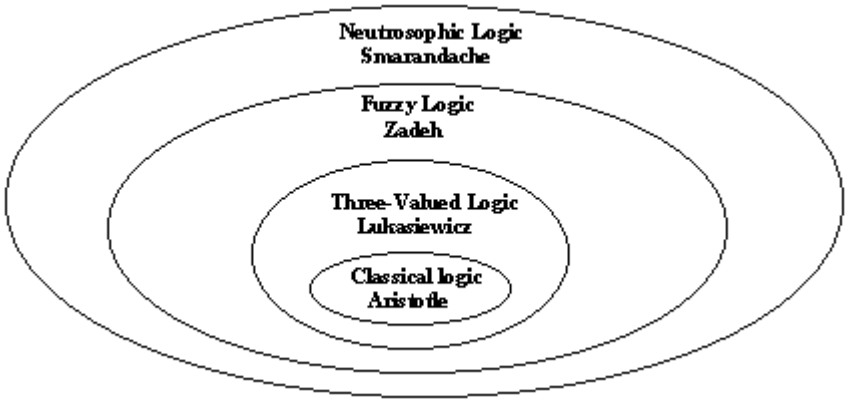
INTRODUCTION TO NEUTROSOPHIC LOGIC



American Research Press
Rehoboth
2002

Charles Ashbacher

Introduction to Neutrosophic Logic



American Research Press
Rehoboth
2002

This book can be ordered in a paper bound reprint from:

Books on Demand
ProQuest Information & Learning
(University of Microfilm International)
300 N. Zeeb Road
P.O. Box 1346, Ann Arbor
MI 48106-1346, USA
Tel.: 1-800-521-0600 (Customer Service)
http://wwwlib.umi.com/bod/

Copyright 2002 by American Research Press and Charles Ashbacher
Rehoboth, Box 141
NM 87322, USA.

Many books can be downloaded from:
http://www.gallup.unm.edu/~smarandache/eBooks-otherformats.htm

This book has been peer reviewed and recommended for publication by:

1) Dr. Jean Dezert
Office National d'Etudes
et de Recherches Aérospatiales (ONERA)
29, Avenue de la Division Leclerc
92320 Chantillon, France.

2) Dr. M Khoshnevisan
Griffith University, Gold Coast
Queensland, Australia.

3) Professor Emeritus C. Corduneanu
Texas State University
Department of Mathematics
Arlington, Texas 76019, USA.

ISBN: 1-931233-60-8

ISBN-13: 978-1-931233-60-6

Standard Address Number: 297-5092

Printed in the United States of America

Foreword

As someone who works heavily in both math and computers, I can truly appreciate the role that logic plays in our modern world. One cannot understand the foundations of mathematics while lacking knowledge of the basics of logic and how proofs are constructed. Two of the first classes I took as a graduate student in mathematics were in the foundations of mathematics, and hardly a day goes by where I do not use some topic from those courses.

Logic is also a fundamental component of advanced computer classes. I am currently teaching advanced courses in assembly language programming and computer organization. Reference is constantly being made to how the rules of logic are incorporated into the fundamental circuits of a computer. The logic used in these classes is known as classical or Boolean logic.

Neutrosophic logic is an extension of classical logic, but as you will see in the book, there are two intermediate steps between them. Neutrosophic logic is yet another idea generated by Florentin Smarandache, who seems to be a perpetual idea machine. Like classical logic, it can be used in many ways, everywhere from statistics to quantum mechanics.

Neutrosophy is more than just a form of logic however. There are several different definitions, extending into many different fields. For our purposes here, we will concentrate almost exclusively on the logic, as the primary purpose of this book is to contrast it with other forms of logic. There is a journal devoted to the publication of papers derived from the ideas of Smarandache called Smarandache Notions Journal that is edited by Dr. Minh Perez. Information about it and other advances can be found on the web site devoted to the posting of the latest results concerning the ideas of Smarandache: http://www.gallup.unm.edu/~smarandache/.

To view some of the latest results in the area of Neutrosophy, go to the sites:
http://www.gallup.unm.edu/~smarandache/NeutrosophicProceedings.pdf
and
http://www.gallup.unm.edu/~smarandache/eBook-Neutrosophics2.pdf.

First International Conference on Neutrosophy, Neutrosophic Logic, Neutrosophic Set, Neutrosophic Probability and Statistics, was held at The University of New Mexico, Gallup, 1-3 December 2001; see the site:
http://www.gallup.unm.edu/~smarandache/FirstNeutConf.htm.

Chapter 1 of this book is devoted to an introduction to the fundamentals of classical logic, the behavior of the connectives as well as the principles of formal reasoning. The next chapter is an examination of an extension of the classical logic called three-valued logic. With values that can be interpreted as “yes”, “no”, and “something else” it is then possible to avoid the restrictions forced on us by the law of the excluded middle. Chapter three describes the rules of fuzzy logic, where the values of the variables can be any value in the continuum from zero to one. Finally, chapter four makes an introduction to neutrosophic logic, where the relationship between it and the other logics will be clear.

As always, the creation of a book is a complex event, from the first germ of an idea to the last spot of glue in the binding. First and foremost, I would like to thank Florentin Smarandache for all of his ideas that I have used in my work. This includes three books, over twenty research papers and several problems that were posed in mathematics journals. He is truly a Renaissance man, I have books of his art, poetry, philosophy and mathematics among my collection. Yes, the plural can be applied to each category several times over.

I would also like to thank the people at American Research Press for their diligence in getting this book into a physical form. Despite all their help, all errors that remain in the book are of course solely my responsibility. Particular praise needs to be given to Minh Perez, who keeps the ideas moving in both directions. Lamarr Widmer of Messiah College read an initial draft of this manuscript and provided many helpful suggestions in how it could be improved. Thanks Lamarr! He and I are both on the editorial board of Smarandache Notions Journal and he is also on the editorial board of Journal of Recreational Mathematics, which I co-edit.

I also would like to once again thank those who have had a hand in my education. My mentor, friend and debating partner Leo Lim is retiring at the end of this academic year after over thirty years of teaching chemistry at Mount Mercy College. They will find it difficult to replace him, and not just because his frame is imprinted into the furniture.

Finally, I would like to mention my family. I have three of the hardest working children around: Katrina, Steven and Becca. At the ages of ten and eleven, they have their own lawn care and snow removal business. They are permanent fixtures in the neighborhood pushing or pulling something. My wife Patti also does a great deal to assist me in my professional endeavors. In most cases, it is nothing more than simply leaving me alone so that I can move my fingers over the keyboard. She also did the cover art for the book.

June, 2002

Charles Ashbacher
Charles Ashbacher Technologies
Box 294, 118 Chaffee Drive
Hiawatha, IA 52233, USA.

Table of Contents

Preface	3
Table of Contents	5
Chapter 1 Classical Logic	7
Section 1 Propositions	7
Section 2 The Law of the Excluded Middle	13
Section 3 Logical Equivalence	14
Section 4 Well-Formed Formulas or WFFs	15
Section 5 An Axiom System For Propositions	16
Section 6 Additional Rules of Inference	17
Section 7 Formal Reasoning	19
Section 8 Quantification Theory	19
Section 9 Uses of Logic In Computer Programming	22
Chapter 2 Three-valued logic	23
Section 1 Lukasiewicz Three-Valued Logic	23
Section 2 The Strong Kleene Three-valued Logic	25
Section 3 The Bochvar Three-valued Logic	27
Section 4 Three-valued Logic In Computer Programming	28
Section 5 Three-valued Logic With an Indeterminate Value	29
Section 6 Properties of Three-Valued Logic	30
Section 7 Modus Ponens in Lukasiewicz Three-Valued Logic	32
Section 8 Rules of Inference in Lukasiewicz Three-Valued Logic	34
Section 9 Tautologies and Contradictions in Three-valued Logic	36
Chapter 3 Fuzzy Logic	37
Section 1 Definition of the Basic Connectives In Fuzzy Logic	37
Section 2 Other Connectives in Fuzzy Logic	39
Section 3 Tautologies and Contradictions In Fuzzy Logic	40
Section 4 Implementing the Fuzzy Connectives in a Computer Program	40
Section 5 Rules of Inference in Fuzzy Logic	46
Section 6 Modal Logic with Fuzzy Variables	48
Section 7 Temporal Logic	50

Chapter 4 Neutrosophic Logic	52
Section 1 Definition of Neutrosphic Logic	52
Section 2 Logical Connectives in Neutrosophic Logic	59
Section 3 Algebraic Properties of Neutrosophic Logics	72
Section 4 Defining Other Connectives in Neutrosophic Logic	95
Section 5 Implementing the Neutrosophic Connectives in Computer Programs	99
Section 6 Ordering The Elements of Neutrosophic Logic	119
Section 7 Rules Of Inference in NL1	123
Section 8 Formal Theories in Neutrosophic Logic	127
Section 7 Reasoning in Neutrosophic Logic	130
References	138
Index	139

Chapter 1

Classical Logic

Section 1
Propositions

In classical logic, a logical variable is restricted to the values of true(T) and false(F). The logical connectives of **and** (∧), **or** (∨) and **not** (¬) in classical logic have the behaviors that are summarized in the truth values of table 1.

Table 1

p	q	p ∧ q	p ∨ q	¬ p
T	T	T	T	F
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

Other names for these connectives are **conjunction** (∧), **disjunction** (∨) and **negation** (¬).

Note: Some authors use the lowercase letters p, q and r and so forth to represent variables that can be either true or false. Others use the uppercase letters A, B, C and so forth. In this book, both will be used.

Since each variable in classical logic is restricted to these two values, if an expression has n different variables, the truth table will have 2^n rows. The result column of a table defines a **Boolean function**, named after George Boole, the mathematician who first described many of the rules of logic. Therefore, a Boolean function is a function that assigns values of true and false to a set of variables and returns a value of true or false.

Definition 1.1.1: Expressions built from variables in the classical logic using these connectives are known as **propositions**.

Examples:
EACH OF THE FOLLOWING ARE PROPOSITIONS

$(p \wedge q) \vee ((r \wedge \neg s) \vee t)$ $(p \vee q \vee r \vee s)$

The evaluation of propositions is done by applying the following simple set of rules:

a) Operations in parentheses are done first with the innermost taking precedence.
b) The unary not connective (¬) is done before the ∧ and ∨.
c) The ∧ and ∨ connectives are considered to be on the same level.

d) If connectives are at the same level, absent parentheses, they are evaluated in the order encountered when you move from left to right.

Example:
Given the proposition

$$p \wedge q \vee (r \wedge \neg s) \vee p$$
$$3 \quad 4 \quad 2 \; 1 \quad 5$$

The order of evaluation of the connectives is marked by the numbers under the connectives. Therefore, the value of the proposition is demonstrated in the truth table of table 2, where the result column is the one above the five.

Table 2

p	q	r	s	p ∧ q ∨ (r ∧ ¬ s) ∨ p
T	T	T	T	T T F F T T
T	T	T	F	T T T T T T
T	T	F	T	T T F F T T
T	T	F	F	T T F T T T
T	F	T	T	F F F F T T
T	F	T	F	F T T T T T
T	F	F	T	F F F F T T
T	F	F	F	F F F T T T
F	T	T	T	F F F F F F
F	T	T	F	F T T T T F
F	T	F	T	F F F F F F
F	T	F	F	F F F T F F
F	F	T	T	F F F F F F
F	F	T	F	F T T T T F
F	F	F	T	F F F F F F
F	F	F	F	F F F T F F
				3 4 2 1 5

Since there are two possible values for every entry in the result column, there are 2^n different Boolean functions for n logical variables.

Given the truth values in the column above the 5, a proposition that defines the Boolean function is

$$p \wedge q \vee (r \wedge \neg s) \vee p$$

Clearly, given any proposition, it is possible to determine the truth values of the Boolean function it defines.

Note: For any Boolean function defined by a truth table, there are many different propositions that can be used to define it.

Given any Boolean function defined by a truth table, it is always possible to construct a proposition whose values match those of the truth table, and this is the topic of the following theorem.

Theorem 1.1.1: Given any Boolean function defined by a truth table, it is possible to construct a proposition using the connectives { $\wedge$, $\vee$, $\neg$ }, whose truth values match the truth table.

Proof: We split it into two cases.

Case 1: The result column contains only F.
In this case, we simply use the expression F, which has the value F for any choice of values for any set of variables.

Case 2: The result column contains at least one T.
Identify all rows where the result is T. For each of these rows, construct an expression by examining the values of the variables. If the variable has the value T, then we use it as is and if the variable has the value F, we place a not operator in front of it. The expression is then built by putting the $\wedge$ connective between the variables.

Example:

Given the row of the truth table

p_1	p_2	p_3	p_4
T	F	F	T

the expression for this row would be

$p_1 \wedge \neg p_2 \wedge \neg p_3 \wedge p_4$

After a moment of thought, it should be clear that this expression is true only for this specific set of values for the variables.

Repeating this for every row where the result is T, we would have a set of expressions, each of which is true for only one assignment to the values and false otherwise. To complete the creation of the total expression, each of these are then put together using the $\vee$ connective. The constructed expression will then have a value of T whenever the result was T and F everywhere else.

Example:
Given the truth values in table 3

Table 3

p	q	r	Result
T	T	T	F
T	T	F	T
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	T
F	F	F	F

the corresponding expression would be

$(p \wedge q \wedge \neg r) \vee (p \wedge \neg q \wedge r) \vee (\neg p \wedge q \wedge r) \vee (\neg p \wedge \neg q \wedge r)$

Definition 1.1.2: The expression that is created using the techniques of theorem 1.1.1 is called the **disjunctive normal form** (**dnf**).

Definition 1.1.3: Given any Boolean function defined by a truth table, if a set of connectives can be used to construct a proposition whose values match that table, then the set of connectives is said to be **complete**. From theorem 1.1.1 it follows that $\{ \wedge, \vee, \neg \}$ is a complete set of connectives.

There are many other binary connectives that can be defined. Of course, given the completeness of the set CS = $\{ \wedge, \vee, \neg \}$ all of them can be constructed using these three operators. Therefore, each of these additional connectives will be considered an abbreviation for an expression built from operators in CS.

Other complete sets of connectives also exist. Proving that a set of connectives is complete is generally done by showing that it is possible to create expressions whose behavior is equivalent to each of the $\vee$, $\wedge$ and $\neg$ connectives.

The definition of the conditional or implication ($\rightarrow$) connective is given in table 4.

Table 4

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

which is equivalent to the truth table of ¬ p ∨ q, as can be verified by examining the truth table for ¬ p ∨ q.

Definition 1.1.4: In the expression p → q, p is known as the **antecedent** and q the **consequence**. The implication is often described as the if-then connective.

The biconditional (↔) connective has the truth values of table 5.

Table 5

p	q	p ↔ q
T	T	T
T	F	F
F	T	F
F	F	T

And p ↔ q is equivalent to the expression (p ∧ q) ∨ (¬ p ∧ ¬q). The ↔ connective can also be considered logical equality.

Exclusive or (^) can be considered logical inequality and has the truth values of table 6.

Table 6

p	Q	p ^ q
T	T	F
T	F	T
F	T	T
F	F	F

And p ^ q has the same truth table as the expression (p ∧ ¬q) ∨ (¬p ∧ q).

Two additional connectives that are commonly used are the joint denial (↓) and the alternative denial (|). The actions of these connectives are summarized in table 7.

Table 7

p	q	p↓ q	p | q
T	T	F	F
T	F	F	T
F	T	F	T
F	F	T	T

The joint denial is also called the NOR connective, an acronym for Not OR. If you examine the truth values it is easy to see that the results are the negations of those for the OR. Alternative denial is also known as the NAND connective, an acronym for Not AND and it is the negation of the AND.
Since $p \downarrow q$ has the same truth table as $\neg p \wedge \neg q$ and $p \mid q$ the same truth table as $\neg p \vee \neg q$, these operators can also be considered abbreviations for more complex expressions. In any case, we are led back to the basic principle that in the classical logic of propositions, anything you wish to express can be done using the elements of CS.

Theorem 1.1.2:
i) $\{ \downarrow \}$ is a complete set of connectives.
ii) $\{ \mid \}$ is a complete set of connectives.

Proof: For each connective, we need a way to represent the behavior of each of the connectives in $\{ \neg, \wedge, \vee \}$.

The behavior of the $\neg$ connective for both is displayed in table 8.

Table 8

p	$p \downarrow p$	$p \mid p$
T	F	F
F	T	T

The behavior of the $\wedge$ connective for both is displayed in table 9.

Table 9

p	q	$(p \downarrow p) \downarrow (q \downarrow q)$	$(p \mid q) \mid (p \mid q)$
T	T	T	T
T	F	F	F
F	T	F	F
F	F	F	F

The behavior of the $\vee$ connective for both is displayed in table 10.

Table 10

P	q	$(p \downarrow q) \downarrow (p \downarrow q)$	$(p \mid p) \mid (q \mid q)$
T	T	T	T
T	F	T	T
F	T	T	T
F	F	F	F

Since the behavior of each of the ↓, | connectives can singularly be used to create expressions with the behavior of each of the ¬, ∧ and ∨ connectives, then each is individually a complete set of connectives.

Definition : A proposition that is true for all values of the variables in the expression is known as a **tautology**. If a proposition is false for all values of the variables in the expression, it is known as a **contradiction**. Note that a proposition p is a tautology if and only if ¬p is a contradiction. If a proposition is neither a tautology or a contradiction, it is called a **contingency**.

Examples:
The following are tautologies:

$A \vee \neg A$ $\quad (A \wedge B) \vee T$ $\quad (A \wedge B) \rightarrow A$
$A \rightarrow (A \vee B)$ $\quad \neg A \rightarrow (A \rightarrow B)$ $\quad (A \wedge B) \rightarrow (A \rightarrow B)$
$\neg(A \rightarrow B) \rightarrow A$ $\quad \neg(A \rightarrow B) \rightarrow \neg B$

and the following are contradictions:

$A \wedge \neg A$ $\quad A \wedge F$

as well as the negations of all the tautologies.

Section 2
The Law of the Excluded Middle

While propositions can be used in many different circumstances, there is a fundamental limitation in their use. Since each expression must be assigned a value that is either true or false, the options are limited. This is known as the **Law of the Excluded Middle**, meaning of course that there is no middle between the two "extreme" values of true and false.

One consequence of this law is the concept of a **vacuous proof**. What this means is that if it is not possible to prove that a valid expression has one value, then it must have the other. In using propositions, if the expression cannot be proven false, then it is considered true. Much like life, in that if you cannot prove that a person is a liar, then you are forced to consider them to be telling the truth.

The vacuous proof appears in the assignment of values to the → connective. It is interpreted as a statement that if the antecedent is true, then the consequence is also true. The statement is then false if the antecedent is true, but the consequence is false. With this notion, if it is not possible to prove the statement false, by the law of excluded middle, it must therefore be true. Hence, the last two rows of the truth table where the antecedent is false have a value of true.

Section 3
Logical Equivalence

Definition 1.3.1: The propositions p and q are said to be **logically equivalent** if $p \leftrightarrow q$ is a tautology. The notation for this relationship is $p \Longleftrightarrow q$. It is possible to show that two expressions are logically equivalent by comparing the entries in the truth table.

Examples: It is easy to verify each of the following logical equivalences

$(p \rightarrow q) \Longleftrightarrow (\neg p \vee q)$ $\qquad$ $(p \leftrightarrow q) \Longleftrightarrow ((p \wedge q) \vee (\neg p \wedge \neg q))$

$(p \uparrow q) \Longleftrightarrow ((p \wedge \neg q) \vee (\neg p \wedge q))$ $\quad$ $(p \downarrow q) \Longleftrightarrow (\neg p \wedge \neg q)$

$(p \mid q) \Longleftrightarrow (\neg p \vee \neg q)$.

The equals sign (=) is often used as an alternate notation for logically equivalent.

The statements in theorem 1.3.1 are all algebraic properties of propositions, where the equals sign is used in place of the arrow. The verifications of the formulas are left as exercises.

Theorem 1.3.1: If A, B and C are propositions:

a) $A \vee B = B \vee A$. Commutativity of $\vee$.
b) $A \wedge B = B \wedge A$. Commutativity of $\wedge$.
c) $(A \wedge B) \wedge C = A \wedge (B \wedge C)$. Associativity of $\wedge$.
d) $(A \vee B) \vee C = A \vee (B \vee C)$. Associativity of $\vee$.
e) $\neg(A \vee B) = \neg A \wedge \neg B$. DeMorgan's rule
f) $\neg(A \wedge B) = \neg A \vee \neg B$. DeMorgan's rule
g) $\neg\neg A = A$. Double negation.
h) $A \wedge A = A$.
i) $A \vee A = A$.
j) $A \wedge \neg A = F$.
k) $A \vee \neg A = T$.
l) $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$. Distributive property.
m) $A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$. Distributive property.
n) $(A \vee B) \wedge A = A$. Absorption law.
o) $(A \wedge B) \vee A = A$. Absorption law.
p) $A \vee F = A$. Identity law.
q) $A \wedge T = A$. Identity law.
r) $A \wedge F = F$. Domination law.
s) $A \vee T = T$. Domination law.

Proof: Left as a set of exercises.

All binary connectives are left associative, meaning that the connectives are evaluated from left to right. For example, the expressions

$A \wedge B \wedge C \wedge D$ and $A \rightarrow B \rightarrow C \rightarrow D$

are evaluated as

$(((A \wedge B) \wedge C) \wedge D)$ and $(((A \rightarrow B) \rightarrow C) \rightarrow D)$.

To evaluate expressions containing several different connectives, first convert all instances of $\rightarrow$, $\leftrightarrow$, ^, $\downarrow$ and | to their logical equivalents in terms of $\wedge$, $\vee$ and $\neg$. The $\wedge$ and $\vee$ connectives are considered to be at the same hierarchical level, so it is necessary to use parentheses if the order of evaluation is to be different from the left to right.

Section 4
Well-Formed Formulas of WFFs

Definition 1.4.1: An expression in classical logic is said to be **well-formed** or a **well-formed formula (wff)** if it can be constructed using the following set of rules:

a) T and F are well-formed.
b) If $\{ p_1, p_2, \ldots, p_k, \ldots \}$ are logical variables restricted to the values of T or F, then all p_I are well-formed.
c) If A is well-formed, then so is (A).
d) If A is well formed, then so is $\neg A$.
e) If A and B are well-formed, then so are $A \wedge B$ and $A \vee B$.
f) Only expressions that can be formed using properties (a) – (e) are well-formed.

The set of propositions is formally defined as all expressions that can be formed using the rules of the definition of a wff.

Understand that if the values of all the variables are known, then the evaluation of the logical connectives will reduce the expression to either T or F.

Note: The connectives $\rightarrow$, $\leftrightarrow$, ^, $\downarrow$ and | do not appear in the above definition of a wff. Therefore, the assumption is that those connectives are replaced by their logically equivalent formulas using $\neg$, $\wedge$ and $\vee$. This is the standard definition of wffs, although it would not alter anything if line (e) is changed to

If A and B are well-formed, then so are $A \wedge B$, $A \vee B$, $A \rightarrow B$, $A \leftrightarrow B$, $A \downarrow B$, A | B and A ^ B.

Section 5

An Axiom System For Propositions

While truth tables are invaluable in working through many of the features of classical logic, they do possess natural limitations. When more general reasoning systems are constructed, it is impractical or impossible to use truth tables to perform the computations. For that we need a **formal theory**.

Definition 1.5.1: A **formal theory** is a system S constructed from the following parts:

a) A set of valid symbols are given as the symbols of S. This set can be either finite or infinite.
b) There is a set of rules defining the well-formed expressions that can be constructed using the symbols of S.
c) A set of wffs is separated out from the complete set of wffs and are called the axioms of S. These expressions are taken to be true by assumption.
d) There is a finite set of relations $R_1, R_2, \ldots, R_n$ between sets of the wffs in S, called the rules of inference. These rules are used to construct proofs, where the statements are:

Given that this set of wffs true, we can conclude that another wff p is also true.

In a proof, the given set of wffs are known as the hypotheses and p the consequence. The complete structure of hypotheses, any intermediate conclusions and the final one is known as a **theorem** in the system S.

Example:
The following is a formal axiomatic theory L

a) The set of symbols in L are { ¬, →, (,), T, F, p_1, p_2, . . . }, where p_I are logical variables.
b) (1) All symbols in { T, F, p_1, p_2, . . . } are wffs in L.
   (2) If A and B are wffs in L, then so are (A), ¬A, and A → B.
   (3) Only expressions that can be formed using rules (1) and (2) are wffs in L.
c) If A and B are wffs in L, then the following are the axioms of L.

(A1) (A → (B → A))
(A2) ((A → (B → C)) → ((A → B) → (A → C)))
(A3) ((¬ B → ¬ A) → ((¬B → A) → B))

d) The only rule of inference in L is **modus ponens**, where B is a consequence of the two hypotheses, $A \rightarrow B$ and A. In other words, if $A \rightarrow B$ and A are true, we can infer that B is also true. This rule is commonly abbreviated MP and the notation used is

$$\frac{A \rightarrow B,\ A}{B}\ .$$

Definition 1.5.2: The formal axiomatic theory L in the previous example is known as the **propositional calculus**.

Once a formal theory is defined, it can then be used to reason from expressions already proven to conclude that additional expressions are also true. Such inferences are called proofs in the theory.

Example:
The following is a proof in L.

Hypothesis: A

Proof:

1) A given
2) $(A \rightarrow (B \rightarrow A))$ Axiom 1

Conclusion by modus ponens $(B \rightarrow A)$.

Note: A proof in a formal scheme like the propositional calculus is very different from the notion of a proof in other areas of mathematics. In general formal proofs are more precise and sequential, with much less latitude in the techniques that you can use.

Section 6
Additional Rules of Inference

Modus ponens is not the only rule of inference that can be used in formal proofs. The following is a list of additional rules that can be applied.

- And – elimination

From a conjunction, any of the propositions in the conjunction can be inferred.

$$\frac{p_1 \wedge p_2 \wedge \ldots \wedge p_r}{p_i}$$

- And – introduction

From a list of propositions, a conjunction can be formed

$$\frac{p_1,\ p_2, \ldots,\ p_r}{p_1 \wedge p_2 \wedge \ldots \wedge p_r}$$

- Or-introduction

IF A PROPOSITION IS TRUE, THEN THE DISJUNCTION WITH IT AND ANY OTHER PROPOSITIONS CAN BE INFERRED.

$$\frac{p_I}{p_1 \vee p_2 \vee \ldots \vee p_r}$$

- Unit resolution

If a disjunction is true and one of the elements is false, then the other must be true.

$$\frac{p \vee q, \neg q}{p}$$

- Modus tollens

Given an implication and the negation of the consequence, we can infer the negation of the antecedent.

$$\frac{p \rightarrow q, \neg q}{\neg p}$$

- Resolution

Implication is transitive

$$\frac{p \rightarrow q, q \rightarrow r}{p \rightarrow r}$$

Section 7
Formal Reasoning

Definition 1.7.1: Let $\Gamma = \{ a_1, a_2, \ldots, a_k \}$ represent a set of wffs. A wff p is said to be a **consequence of Γ** if it is possible to construct a proof of p that starts with the wffs of Γ. The elements of Γ are called the hypotheses or premises and p is the consequence. This can be considered as the logical operation of starting from a set of statements known to be true and using the inference rules of the system to conclude that another statement is also true. The notation for such an inference is $\Gamma \vdash p$. Generally, we write

$a_1, a_2, \ldots, a_k \vdash p.$

rather than

$\{ a_1, a_2, \ldots, a_k \} \vdash p.$

The following are all direct consequences of the definition of consequence.

a) $a_1, a_2, \ldots, a_k \vdash a_i.$

b) If $a_1, a_2, \ldots, a_k \vdash p$ then $a_1, a_2, \ldots, a_k, b_1, \ldots, b_j \vdash p$, where b_I are also wffs.

In other words, if we can prove a wff p using the wffs in a set, adding additional wffs to the hypotheses does not alter the ability to derive p.

Section 8

QUANTIFICATION THEORY

While propositions allow us to do a great deal of work, they are limited in their general use because it is not possible to use them to represent many expressions. Consider the following sentences:

i) Any friend of yours is a friend of mine.
ii) The enemy of my enemy is my friend.
iii) All men are mortal.

These expressions contain within them features where the truth value of the expression is dependent on internal variables and the relationships between the features. For example, in the first expression, there could be many different people considered to a friend of yours. To determine if this expression is true, we would have to take each and every instance of friend of yours and determine if they are also a friend of mine. This also requires an understandable definition of the word friend.

Definition 1.8.1: Words such as all, any, every, and some are used to place restrictions on the use of a variable. The formal term for this is **quantify**, and the logical system is called **quantification theory**.

Example:
In the statement

All men are mortal

The word "All" of course means that the expression holds for every value for the variable men. The formal mathematical term is "for all" and it is known as the **universal quantifier**. If P(x) is the assertion that x has the property P, then (x)P(x) means that for all x, P(x) is true. If no element satisfies the property, then the expression would be (x) ¬ P(x) and if at least one element does not satisfy the property, the expression is ¬(x)P(x).

In the case where at least one element satisfies the property, the **existential quantifier** can be used. The word equivalent is "there exists" and the expresssion is (Ex)P(x), which means there exists an x such that P(x) is true.

Note: Only one of the existential or universal quantifiers need be defined. For example, if we have the existential quantifier, we can write the expression ¬((Ex)¬P(x)) which means, it is not the case that there is some x such that P(x) is false. Since there is no x that makes P(x) false, it must be true for all x, which is equivalent to (x) P(x).

Similarly, if we have the universal quantifier, we can write the expression ¬((x) ¬P(x)) which means, it is not the case that for all x, P(x) is false. Since P(x) is not false for all x, it must be true for some x.

Definition 1.8.2: When using quantifiers, the set of all values that the variables can have is known as the **universe of discourse**.

Definition 1.8.2: An expression that contains a quantifier is called a **predicate**.

Definition 1.8.3: If a variable is acted on by a quantifier, it is said to be **bound**. A variable that is not acted on by a quantifier is said to be **free**.

Example:
In the expression

(Ex) x > y

x is a bound variable and y is a free variable. Since y is free, it can be replaced by any other variable name without altering the meaning of the expression. Therefore, the expression

(Ex) $x > z$

is mathematically equivalent.

Once the values of the elements in the universe of discourse is known, it is possible to determine whether the predicate is true or false if it contains no free variables.

Example:
If the universe of discourse is the set of numbers U = { 1, 2, 3, 4, 5 }, then the expression

(Ex) $x > 0$

is true. Furthermore, the expression

(x) $x > 0$

is also true.

Quantification theory is often used with infinite sets of discourse. In these cases, it is necessary to use theory to determine if the expression is true or false.

Example:
If the universe of discourse is the set of all integers, then

(n) $n^2 > 0$ is false
(n) $n^2 \geq n$ is true
(E n) $n^2 = 2$ is false

The English equivalents of these predicates are

The squares of all integers are greater than zero.
The squares of all integers n are greater than or equal to n.
There is an integer that is the square root of two.

Example:
If the universe of discourse is the set of all real numbers x where $0 \leq x \leq 1.0$, then

(x) $x^2 \leq 1.0$ is true
(x) $x^2 \leq x$ is true
(E x) $x^2 < 0$ is false

The English equivalents of these predicates are

The square of any real number between zero and one inclusive is less than or equal to one.

The square of any real number between zero and one inclusive is less than or equal to the number.
The square of any real number between zero and one is less than zero.

The connectives of propositions can also be used in combination with predicates.

Example:
If the universe of discourse is the set of all real numbers x where $0 \leq x \leq 1.0$, then

$((x)\ x^2 \leq 1.0) \wedge ((x)\ x^2 \leq x)$ is true
$((x)\ x^2 < 0) \vee ((x)\ x^2 \leq 1.0)$ is true
$((x)\ x^2 < 0) \vee \neg ((x)\ x^2 \leq 1.0)$ is false

Section 9
Uses of Logic In Computer Programming

Logic is a fundamental component of computer programming. Languages such as C++ and Java contain logical operators corresponding to the conjunction, disjunction and negation. They are the double ampersand (&&), double vertical bar (||) and exclamation point (!) respectively. These operators accept values that are Boolean and return a Boolean. The behavior of the and, or and not operations is similar to those of classical logic.

Java and other languages contain additional operators that perform bitwise operations. Data in computers is expressed in binary form, or as a sequence of zeros and ones. A bitwise operation matches the bit positions in two binary strings and performs the operation one position at a time. The operations are similar to those of classical logic, with true and false replaced by one and zero respectively.

Example:
The actions of the bitwise operators are demonstrated in table 11.

Table 11

m	0	1	1	0	1	1	1	0	0	0	1	1	0	0	1	1
n	1	1	1	0	0	0	1	1	0	1	1	0	1	1	1	0
m and n	0	1	1	0	0	0	1	0	0	0	1	0	0	0	1	0
m or n	1	1	1	0	1	1	1	1	0	1	1	1	1	1	1	1
not m	1	0	0	1	0	0	0	1	1	1	0	0	1	1	0	0

CHAPTER 2
THREE-VALUED LOGIC

Section 1
Lukasiewicz Three-Valued Logic

With only two possible values for the variables, classical logic uses what is known as the **Law Of The Excluded Middle**. This simply means that there are only the two extremes of true and false with no middle values possible. While there were some rumblings about possible different types of logic for several centuries, the first to work out a system with more than two truth values was Jan Lukasiewicz, a Polish logician. In this system, there are three possible values, 1, ½ and 0.

The negation operator in Lukasiewicz three-valued logic is defined in table 1.

Table 1

p	¬ p
1	0
1/2	½
0	1

In this logic, the ½ can be considered as an intermediate value of half true and half false.

Another way that the negation could be defined is

$$\neg p = 1 - p.$$

The definitions of the $\vee$ and $\wedge$ connectives in the Lukasiewicz three-valued logic are given in table 2.

Table 2

p	Q	p $\vee$ q	p $\wedge$ q
1	1	1	1
1	½	1	1/2
1	0	1	0
1/2	1	1	1/2
1/2	½	1/2	1/2
1/2	0	1/2	0
0	1	1	0
0	½	1/2	0
0	0	0	0

Note that $p \vee q$ could be defined as $\max\{ p, q \}$ and $p \wedge q$ as $\min \{ p, q \}$.

Using the equivalence formulas $(p \rightarrow q) \Leftrightarrow (\neg p \vee q)$,
$((p \wedge q) \vee (\neg p \wedge \neg q)) \Leftrightarrow (p \leftrightarrow q)$, $((\neg p \wedge q) \vee (p \wedge \neg q)) \Leftrightarrow (p \wedge q)$,
$(\neg p \wedge \neg q) \Leftrightarrow (p \downarrow q)$ and $((\neg p \wedge q) \vee (p \wedge \neg q) \wedge (\neg p \wedge \neg q)) \Leftrightarrow (p \mid q)$ definitions for these additional connectives in the Lukasiewicz three-valued logic are given in table 3.

Table 3

p	q	$p \rightarrow q$	$p \leftrightarrow q$	$p \wedge q$	$p \downarrow q$	$p \mid q$
1	1	1	1	0	0	0
1	1/2	1/2	1/2	1/2	0	1/2
1	0	0	0	1	0	1
1/2	1	1	1/2	1/2	0	1/2
1/2	½	1/2	1/2	1/2	½	1/2
1/2	0	1/2	1/2	1/2	½	1/2
0	1	1	0	1	0	1
0	½	1	1/2	1/2	½	1/2
0	0	1	1	0	1	1

Note that if 1 is considered a representation of true and 0 false, then the row entries for p and q for zero and one are those of classical logic. Therefore, this three-valued logic is an extension of classical logic. In most cases, new mathematical structures are defined so that they are consistent with those previously defined.

The definitions of the connectives in table 3 are consistent with the equivalence formulas in classical logic. It is not necessary to maintain this consistency for all logic structures that are extensions of the classical logic. For example, the definition of the $\rightarrow$ connective can be changed to that in table 4.

Table 4

P	q	$p \rightarrow q$
1	1	1
1	½	1/2
1	0	0
1/2	1	1
1/2	½	1/2
1/2	0	0
0	1	1
0	½	1
0	0	1

This definition is still consistent with classical logic, as can be seen by examining all rows where the values of p and q are zero or one.

A third definition of the → connective can be used, where the additional rule is that if p > q, then p → q = 0. The truth table for this definition of the → connective is given in table 5.

Table 5

P	q	p → q
1	1	1
1	½	0
1	0	0
½	1	1
½	½	1/2
½	0	0
0	1	1
0	½	1
0	0	1

This definition is also consistent with the classical definition of the implication connective.

To complete the examination of all connectives, Lukasiewicz defined the implication and biconditional connectives in the manner illustrated in table 6.

Table 6

p	q	p → q	p ↔ q
1	1	1	1
1	½	½	1/2
1	0	0	0
1/2	1	1	1/2
1/2	½	1	1
1/2	0	½	1/2
0	1	1	0
0	½	1	1/2
0	0	1	1

Section 2
The Strong Kleene Three-valued Logic

Another form of three-valued logic was developed by S. Kleene. The third value in this logic is U, which is interpreted as undefined or unknown and the truth tables of the and, or and not connectives are demonstrated in tables 7 and 8.

Table 7

p	q	$p \vee q$	$p \wedge q$	$p \rightarrow q$	$p \leftrightarrow q$
T	T	T	T	T	T
T	F	T	F	F	F
T	U	T	U	U	U
F	T	T	F	T	F
F	F	F	F	T	T
F	U	U	F	T	U
U	T	T	U	T	U
U	F	U	F	U	U
U	U	U	U	U	U

Table 8

p	$\neg p$
T	F
F	T
U	U

The truth tables of the Kleene three-valued logic can be rewritten using an I for indeterminate rather than a U for unknown.

Note: The definition of the connectives in the strong Kleene three-value logic are also an extension of the classical logic.

Note: Since the values in this logic are not numeric, the definitions of the connectives cannot involve any arithmetic operations such as max and min. This can be a limitation in the way the connectives are used.

Note: The remaining connectives for strong Kleene logic could be defined using the logical equivalences of classical logic.

It is possible to define universal and existential quantifiers in the strong Kleene logic. This is done by treating universal quantification as an infinite conjunction and the existential quantifier as an infinite disjunction.

Definition 2.2.1: Let

$$\bigwedge_{i \varepsilon U} p_i$$

be an infinite conjunction of variables in the strong Kleene logic. It has the value T if all p_I are T, F if some p_I is F and U otherwise.

Definition 2.2.2: Let

$$\bigvee_{i \varepsilon U} p_i$$

be an infinite disjunction of variables in the strong Kleene logic. It has the value T if some p_I is T, F if all p_I are F and U otherwise.

Section 3
The Bochvar Three-valued Logic

An additional three-valued logic was created by D. Bochvar and was inspired by the examination of semantic paradoxes. For example, there is the classic statement, "This expression is false." If it is true, then it must be false and if it is false, then it must be true, which is a paradox. Bochvar's solution to such statements is to introduce the additional value M, which represents meaningless or paradoxical. The behavior of the connectives are simple, in that if any variable in the expression has the value M, then the expression has the value M. This is demonstrated in tables 9 and 10.

Table 9

9	¬ p
T	F
F	T
M	M

Table 10

p	Q	p ∨ q	p ∧ q	p → q	p ↔ q	p^q	p↓q	p|q
T	T	T	T	T	T	F	F	F
T	F	T	F	F	F	T	F	T
T	M	M	M	M	M	M	M	M
F	T	T	F	T	F	T	F	T
F	F	F	F	T	T	F	T	T
F	M	M	M	M	M	M	M	M
M	T	M	M	M	M	M	M	M

M	F	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M

Note: Once again, the connectives defined for Bochvar's logic are extensions of their classical counterparts.

Note: The Bochvar three-valued logic can be considered one where the M is dominant.

It is possible to define universal and existential quantifiers in the Bochvar's logic as well. Once again, infinite conjunctions and disjunctions are used.

Definition 2.3.1: Let

$$\bigwedge_{i \in U} p_i$$

be an infinite conjunction of variables in the Bochvar's logic. It has the value T if all p_I are T, F if some p_I is F and no variables are M and M otherwise.

Definition 2.3.2: Let

$$\bigvee_{i \in U} p_i$$

be an infinite disjunction of variables in the strong Bochvar's logic. It has the value T if some p_I is T and no variables are M, F if each p_I is F and M otherwise.

Section 4
Three-valued Logic In Computer Programming

In programming languages such as C, C++ and Java, the and (&&) and or (||) connectives are evaluated in a **conditional** manner. For the and expression

E_1 && E_2

The expression E_1 is evaluated first and if it is false, E_2 is not evaluated and the expression returns an F. Therefore, in this case, it makes no difference if E_2 is undefined, as E_2 is evaluated only when E_1 is T. If E_1 is true, then the result is the value of E_2.

If the expression is an or

E_1 || E_2

The expression E_1 is again evaluated first and if it is true, E_2 is not evaluated and the expression returns a T. E_2 is evaluated only when E_1 is false. The value of the expression

is then E_2 when E_1 is false. The behavior of the conditional or and and evaluations in computing is summarized in table 11.

Table 11

p	q	p && q	p ‖ q
T	T	T	T
T	F	F	T
T	U	U	T
F	T	F	T
F	F	F	F
F	U	F	U
U	T	U	U
U	F	U	U
U	U	U	U

Section 5
Three-valued Logic With an Indeterminate Value

Another form of three-valued logic uses the value I for indeterminate. In this logic, if any variable has the value I, then the result is I. The truth tables would be the same as those for the Bochvar's logic, where the M is replaced by an I.

The definition of three-valued logic that uses the I has applications in the branch of physics known as quantum mechanics. Quantum mechanics is the physics of the very small, where events can simultaneously be in more than one state and the current state is not known until the measurement takes place.

Perhaps the most famous of all quantum mechanical descriptions is the paradox of Shrodinger's cat. A live cat is placed in a closed container with a single atom of a radioactive element. If the atom decays, it will be detected in the container and will cause a vial of poison to be broken, which will kill the cat.

According to quantum mechanics, the probability that the atom will decay during the interval of its' half-life is ½. Therefore, if the cat is placed in the container and it is closed for the interval, according to classical interpretations, the probability that the cat is alive after the interval is ½.

However, in the weird world of quantum mechanics, the atom can decay, but until the box is opened and examined, the cat is in an intermediate state of being neither alive nor dead. Or, for the optimist in you, it can be considered simultaneously alive and dead. Therefore, until the examination takes place, it is inaccurate to assign a value of true or

false to the condition of the cat being alive. While classical logic is of little value here, three-valued logic can represent this situation. By assigning the condition of the cat before the examination the value of I, the state of the cat is accurately described.

As we will see in a later chapter, this also allows for the addition of a **temporal** value, or one that represents the point in time.

Section 6

Properties of Three-Valued Logic

There are several properties that hold in the Lukasiewicz three-valued logic, and some are listed and proven below. Many of them also hold for the other three-valued logic structures, but the numeric representations used in the Lukasiewicz logic make it better suited as preparation for the additional logic structures we will examine in chapters 3 and 4.

Theorem 2.6.1: If A is a variable in the Lukasiewicz three-valued logic, then the following formulas are satisfied:

i) $A \wedge 1 = A$.
ii) $A \wedge 0 = 0$.
iii) $A \wedge ½ \leq ½$.
iv) $A \vee 1 = 1$.
v) $A \vee 0 = A$.
vi) $A \vee ½ \geq ½$.
vii) $A \wedge A = A$
viii) $A \vee A = A$.
ix) $\neg\neg A = A$.

Proof:
i) If A = 1, then $1 \wedge 1 = 1$. If A = ½, then $½ \wedge 1 = ½$ and if A = 0, then $0 \wedge 1 = 0$.
ii) If A = 1, then $1 \wedge 0 = 0$. If A = ½, then $½ \wedge 0 = 0$ and if A = 0, then $0 \wedge 0 = 0$.
iii) If A = 1, then $1 \wedge ½ = ½$. If A = ½, then $½ \wedge ½ = ½$ and if A = 0, then $0 \wedge ½ = 0$.
iv) If A = 1, then $1 \vee 1 = 1$. If A = ½, then $½ \vee 1 = 1$ and if A = 0, then $0 \vee 1 = 1$.
v) If A = 1, then $1 \vee 0 = 1$. If A = ½, then $½ \vee 0 = ½$ and if A = 0, then $0 \vee 0 = 0$.
vi) If A = 1, then $1 \vee ½ = 1$. If A = ½, then $½ \vee ½ = ½$ and if A = 0, then $0 \vee ½ = ½$.
vii) If A = 1, then $1 \wedge 1 = 1$, if A = ½, then $½ \wedge ½ = ½$ and if A = 0, $0 \wedge 0 = 0$.
viii) If A = 1, then $1 \vee 1 = 1$, if A = ½, then $½ \vee ½ = ½$ and if A = 0, $0 \vee 0 = 0$.
ix) For any number x, $1 - (1 - x) = 1 - 1 + x = x$.

Theorem 2.6.2: If A, B and C are variables in the Lukasiewicz three-valued logic:

i) $A \vee B = B \vee A$.
ii) $A \wedge B = B \wedge A$.
iii) $A \vee (B \vee C) = (A \vee B) \vee C$.

iv) $A \wedge (B \wedge C) = (A \wedge B) \wedge C$.

In other words, $\wedge$ and $\vee$ are associative and commutative in three-valued logic.

Proof: To prove these expressions, we will rely on the equivalent definitions where

$A \vee B = \max\{ A, B \}$ $A \wedge B = \min\{ A, B \}$.

i) $A \vee B = \max\{ A, B \} = \max\{ B, A \} = B \vee A$.
ii) $A \wedge B = \min\{ A, B \} = \min\{ B, A \} = B \vee A$.
iii) $A \vee (B \vee C) = \max\{ A, \max\{ B,C \} \} = \max\{ A, B, C \} =$
$\max\{ \max\{ A, B \}, C \} = (A \vee B) \vee C$.
iv) $A \wedge (B \wedge C) = \min\{ A, \min\{ B,C \} \} = \min\{ A, B, C \} =$
$\min\{ \min\{ A, B \}, C \} = (A \wedge B) \wedge C$.

Theorem 2.6.3: If A and B are variables in a three-valued logic

i) $(A \vee B) \wedge A = A$ Absorption
ii) $(A \wedge B) \vee A = A$ Absorption
iii) $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$ Distributive
iv) $A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$ Distributive

Proof:
Once again, these formulas are easy to verify using the definitions

$A \vee B = \max\{ A, B \}$ $A \wedge B = \min\{ A, B \}$.

i) If $\max\{ A,B \} = A$, then the expression is $A \wedge A$ which is A. If $\max\{ A,B \} = B$, then $\min\{ A, B \} = A$.
ii) If $\min\{ A, B \} = A$, then $A \vee A = A$. If $\min\{ A, B \} = B$, then $\max\{ A, B \} = A$.
iii) This proof is done using case analysis.

Case 1: $A \geq B \geq C$
$B \wedge C = C$ and then $A \vee C = A$, so the left side has the value A. $(A \vee B) = A$ and $(A \vee C) = A$, so the right side is $A \wedge A = A$.

Case 2: $A \geq C \geq B$
This is similar to case 1, simply reverse the roles of B and C.

Case 3: $B \geq A \geq C$
$(B \wedge C) = C$ and $A \vee C = A$, so the left side has the value A. $(A \vee B) = B$, $(A \vee C) = A$, and $A \wedge B = A$, so the right side has the value A.

Case 4: $C \geq A \geq B$
$(B \wedge C) = B$ and $A \vee B = A$, so the left side has the value A. $(A \vee B) = A$, $(A \vee C) = C$, and $A \wedge C = A$, so the right side has the value A.

Case 5: $B \geq C \geq A$
$(B \wedge C) = C$ and $A \vee C = C$, so the left side has the value C. $(A \vee B) = B$, $(A \vee C) = C$, and $B \wedge C = C$, so the right side has the value C.

Case 6: $C \geq B \geq A$
This is similar to case 5, simply interchange the roles of B and C.

Theorem 2.6.4: If A and B are variables in a three-valued logic

i) $\neg(A \vee B) = \neg A \wedge \neg B$ DeMorgan's
ii) $\neg(A \wedge B) = \neg A \vee \neg B$ DeMorgan's

Proof:
i) The proof is by case analysis

Case 1:
If A or B is one, then the left side is zero. One of the two negations on the right is zero, so the conjunction on the right is zero.

Case 2:
If the largest value of the variables is ½, then the left side is ½. If both are ½, then each negation on the right is ½ and the conjunction is ½. If one is ½ and the other is 0, then one negation is ½ and the other is 1, so the conjunction is ½.

Case 3: Both values are zero. In this case, the disjunction on the left is zero and the negation is one. Both negations on the right are one, so the conjunction is 1.

iii) The proof is by case analysis

Case 1:
If one of the values is zero, then the conjunction in the left is zero, so the left side is 1. One of the negations on the right is one and the disjunction would be one.

Case 2:
If one of the variables is ½, then the conjunction on the left is ½, so the value of the left is ½. Since one of the negations on the right is ½ and the other is either ½ or zero, the value of the right is also ½.

Case 3: Both variables are one. Then the conjunction on the left is 1, so the value of the left is zero. Both negations on the right are zero, so the disjunction is also zero.

Section 7

Modus Ponens in Lukasiewicz Three-Valued Logic

The inference rule of modus ponens can be interpreted in the following way. If $p \rightarrow q$ and p are true then we can conclude that q is also true. Recall that the truth table of the $\rightarrow$ connective is

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

so MP can be interpreted as meaning that both p and $p \rightarrow q$ being true restricts us to the first line of the table, where q is also true.

MP can be modified to the situation of the Lukasiewicz three-valued logic, only now we have to be more precise in our definitions of "true."

Definition 2.7.1: The MP inference rule can be defined in the following way in three-valued logic.

i) If $p = 1$ and $p \rightarrow q$ is 1 or ½, then q has the value of $p \rightarrow q$.
ii) If $p = ½$ and $p \rightarrow q = 1$, then $q = 1$.
iii) If $p = ½$ and $p \rightarrow q = ½$, then $q = 1/2$.

The cases where $p = 0$ or $p \rightarrow q = 0$ need not be considered, as modus ponens requires that they both be "true".

Note how this coincides with the truth tables for the $\rightarrow$ connective in tables 4 and 5, but not that in table 3. A theorem in the three-valued logic then must also specify the truth values being assigned to the hypotheses as a consequence of which definition of the connectives is being used. Formal theories can also be created in the three-valued logic, the only complication is to add additional material that specifies the constants and makes precise the rules of inference.

Example: The following is a formal theory L3V in the Lukasiewicz three-valued logic.

a) The set of symbols in L3V are { $\neg, \vee, \wedge, \rightarrow$, (,), 1, 0, ½, A, B, . . . }.
b) 1) All symbols in { 1, 0, ½, A, B, . . . } are wffs in L3V.
   2) If A and B are wffs in L3V, then so are (A), $\neg$A, $A \vee B$, $A \rightarrow B$ and $A \wedge B$.
   3) Only expressions that can be formed using rules (1) and (2) are wffs in L3V.
c) If A and B are wffs in L3V, then the following are the axioms of L3V.

1) $A \rightarrow (A \vee B)$ has a value greater than zero.
2) $(A \wedge B) \rightarrow A$ has a value greater than zero.
3) $(A \wedge B) \rightarrow B$ has a value greater than zero.

d) The only rule of inference is modus ponens, where $\rightarrow$ is defined in table 4.

The following are theorems in L3V

a) From A not zero we can infer $A \vee B$ with at least the same truth value as A.

Proof:
1) $(A \rightarrow (A \vee B)) = 1$. L3V axiom 1.
2) A has a value greater than zero. Given.
3) i) If A = 1, then applying MP3V on 1) and 2) we can conclude that $(A \vee B) = 1$.
   ii) If A = ½, then applying MP3V on 1) and 2) we can conclude that $(A \vee B) = ½$.

b) From $A \wedge B$ not zero we can infer A with the same truth value as $A \wedge B$.

Proof:
1) $(A \wedge B \rightarrow A) = 1$ L3V axiom 2.
2) $A \wedge B$ is not zero. Given.
3) i) If $A \wedge B = 1$ then we can apply L3V on 1) and 2) to conclude that A = 1.
   ii) If $A \wedge B = 1/2$ then we can apply L3V on 1) and 2) to conclude that A = 1/2.

Section 8
Rules of Inference in Lukasiewicz Three-Valued Logic

Rules of inference are written in the form

$$\frac{p_1, p_2, \ldots, p_k}{q}$$

The modus ponens rule of inference for classical logic can be written in the form

$$\frac{p \rightarrow q, p}{q}$$

However, this notation must be altered in the realm of three-valued logic, and each inference rule will have a different qualification. To start, the principle will be that all propositions above the line have values greater than zero. Furthermore, the additional qualification will be that the conclusion under the line must have a value greater than or equal to the values of all the expressions above the line. This is expressed in the formal definition.

Definition 2.8.1: A rule of inference

$$\frac{p_1, p_2, \ldots, p_k}{q}$$

is valid in the Lukasiewicz three-valued logic if whenever each of the expressions $p_1, p_2, \ldots, p_k$ is greater than zero, the expression q has a value that is nonzero. At times it is also helpful to clarify the value of q.

For example, from the table of values for the $\rightarrow$ connective in three-valued logic as defined in tables 4 and 5, it is possible to conclude that from $p \rightarrow q$ and p we can conclude q with a value of at least ½. This result is not valid for the definition in table 3.

If it is qualified, the and-elimination rule of inference also can be used in Lukasiewicz three-valued logic,

Given $p_1 \wedge p_2 \wedge \ldots \wedge p_k = v$, then p_I can be inferred to have a value $\geq v$.

The and-introduction rule of inference also has a qualified meaning in three-valued logic.

Given that $p_1, p_2, \ldots, p_k$ are all nonzero, we can infer that $p_1 \wedge p_2 \wedge \ldots \wedge p_k$ is also nonzero.

The or-introduction rule of inference has a qualified meaning in three-valued logic.

Given p, we can infer $p \vee q$ having a value greater than or equal to p.

The unit-resolution rule of inference does not apply in Lukasiewicz three-valued logic, for if $p \vee q$ and $\neg q$ are nonzero, then it is not possible to conclude that p is nonzero. Use the values of p = 0 and q = ½ to verify this.

The modus tollens rule of inference also does not apply in three-valued logic for the $\rightarrow$ connective as defined in table 3. For if p = 1, q = ½, then $p \rightarrow q$ = ½, $\neg q$ = ½ and $\neg p$ = 0. It also does not apply if the definition of the $\rightarrow$ connective is that of table 4. For if p = 1, q = ½, $p \rightarrow q$ = ½ and $\neg p$ = 0. However, the modus tollens rule of inference does apply if the definition of the $\rightarrow$ connective is that of table 5. In all cases where $p \rightarrow q$ and $\neg q$ are nonzero, $\neg p$ is also nonzero.

The resolution rule of inference also applies in Lukasiewicz three-valued logic. For if, $p \rightarrow q$ and $q \rightarrow r$ are nonzero, then $p \rightarrow r$ is also nonzero. This can be verified by the examination of the truth table for the three variables p, q and r and the three different definitions of the $\rightarrow$ connective.

The definition of formal reasoning in Lukasiewicz three-valued logic is similar to the definitions of the inference rules.

Definition 2.8.2: Let $\Gamma = \{ a_1, a_2, \ldots, a_k \}$ represent a set of wffs in Lukasiewicz three-valued logic. A wff p is said to be a **consequence of Γ** if whenever all the elements of Γ are all nonzero, then p is also nonzero.

The notation for a consequence in three-valued logic is the same as that for classical logic.

$$a_1, a_2, \ldots, a_k \vdash p.$$

Section 9
Tautologies and Contradictions in Lukasiewicz Three-valued Logic

The definition of tautologies and contradictions in three-valued logic is a natural one.

Definition 2.9.1: A proposition p in three-valued logic is a tautology if it has the value 1(T) for all values of the variables in p. It is a contradiction if it has the value 0(F) for all values of the variables.

Tautologies and contradictions are much less common in three-valued logic and this is easy to see from the following theorem.

Theorem 2.9.1: No expression in a three-valued logic can be a tautology(contradiction) if it is not also a tautology(contradiction) in classical logic.

Proof: Since three-valued logic restricted to the values of one(T) and zero(F) is classical logic, if an expression is true for all values of the three variables, it must also be true for the variables being zero and one.

Since $p \rightarrow p$, $p \leftrightarrow p$ and $p \vee \neg p$ are tautologies in classical logic but not in Lukasiewicz three-valued logic, there are "fewer" tautologies in Lukasiewicz three valued logic. Since $p \vee 1$ is always 1 and $p \wedge 0$ is always zero, tautologies and contradictions do exist in Lukasiewicz three-valued logic. The other three-valued logics have similar properties.

Depending on the circumstances, we may be interested in values of the expression that are above a certain value rather than precisely one.

Definition 2.9.2: The expression p in the Lukasiewicz three-valued logic is said to be a **one-half tautology** if it has a value greater than or equal to ½ for all values of the variables in p.

These tautologies do exist, as $p \vee \neg p$, $p \rightarrow p$ and $p \leftrightarrow p$ all have values that are greater than or equal to ½ for all values of p.

Chapter 3

Fuzzy Logic

Section 1
Definition of the Basic Connectives In Fuzzy Logic

In the late 60's, Lofti A. Zadeh, a professor in the Department of EE/CS at the University of California at Berkeley, devised an expansion of the classical and multi-valued logics known as **fuzzy logic**. The scheme uses the basic idea from probability that an event can have a probability between 1.0 (certain to happen) and 0.0 (certain not to happen). This gradation of likelihood is applied to logic, creating degrees of truth. Fuzzy logic was originally developed for application in problems of knowledge representation and is a more intuitive system for describing events before they happen. For example, on the day after tomorrow, we will be able to apply a value of T or F to the expression, "It will snow tomorrow." However, today we have no such certainty, but based on experience and the knowledge of weather patterns, it is possible to say, "There is an 80% chance that it will snow tomorrow."

Fuzzy logic allows values between 0.0 and 1.0 to be applied to the variables. If the value is 0.0, then the variable is considered to be false and if the value is 1.0, it is considered to be true. For intermediate values, such as x = 0.67, we use expressions like:

There is a 67% chance that x is true.
Approximately two-thirds of the time x is true.

Fuzzy logic more accurately describes the world we live in. Phrases such as likely, not likely, probably, unlikely, most, few, usually and nearly all are much more common than the absolutes of true and false.

The basic logical operations of CS = { $\wedge$, $\vee$, $\neg$ } have simple definitions in fuzzy logic.

$x \wedge y$ = smallest of the values of x and y.

$x \vee y$ = largest of the values of x and y.

$\neg x = 1.0 - x$.

Examples:
If x = 0.45 and y = 0.84, then $x \wedge y = 0.45$, $x \vee y = 0.84$ and $\neg x = 1.0 - 0.45 = 0.55$.

It should be clear that if the values of the variables are restricted to 1.0 and 0.0, the fuzzy connectives behave the same way as the classical operators with 1.0 equivalent to T and 0.0 equivalent to F. If the restrictions are to the three possibilities 1,.0, 0.0 and 0.5, then the behavior is that of the Lukasiewicz three-valued logic.

Note the similarities in how the $\vee$ and $\wedge$ connectives are defined in fuzzy logic to the definitions in three-valued logic.

Fuzzy logic has found many uses in areas such as control systems, expert systems, inference engines and artificial intelligence. Many devices, such as automobiles, appliances, elevators, trains and toys contain embedded devices that apply the properties of fuzzy logic to define their rules of behavior.

Many of the expressions that were true in the three-valued logic are also true in a fuzzy logic.

Theorem 3.1.1: If A is a variable in a fuzzy logic, then the following formulas are satisfied:

i) $A \wedge 1 = A$.
ii) $A \wedge 0 = 0$.
iii) $A \vee 1 = 1$.
iv) $A \vee 0 = A$.
v) $\neg\neg A = A$.

Proof: The proofs all rely on the definitions of the connectives.
i) Since $A \wedge 1 = \min\{ A, 1\}$ and $A \leq 1$ the result is immediate.
ii) Since $A \wedge 0 = \min \{ A, 0 \}$ and $A \geq 0$, the result is immediate.
iii) Since $A \vee 1 = \max \{ A, 1 \}$ and $A \leq 1$ the result is immediate.
iv) Since $A \vee 0 = \max \{ A, 0 \}$ and $A \geq 0$ the result is immediate.
v) $\neg\neg A = 1 - (1 - x) = 1 - 1 + x = x = A$.

Fuzzy logic can be considered an infinite valued logic. Since the real interval from zero to one is uncountable, the number of possible values for variables in fuzzy logic is uncountably infinite.

Theorem 3.1.2: If A, B and C are variables in fuzzy logic:

i) $A \vee B = B \vee A$.
ii) $A \wedge B = B \wedge A$.
iii) $A \vee (B \vee C) = (A \vee B) \vee C$.
iv) $A \wedge (B \wedge C) = (A \wedge B) \wedge C$.

In other words, $\vee$ and $\wedge$ are commutative and associative in fuzzy logic. Proofs of the elements of the theorem are similar to those of theorem 2.6.2.

Theorem 3.1.3: If A and B are variables in fuzzy logic

i) $(A \vee B) \wedge A = A$ Absorption
ii) $(A \wedge B) \vee A = A$ Absorption
iii) $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$ Distributive

iv) $A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$ Distributive

The proofs of these expressions are similar to those of theorem 2.6.3.

Theorem 3.1.4: If A and B are variables in a fuzzy logic

i) $\neg(A \vee B) = \neg A \wedge \neg B$ DeMorgan's
ii) $\neg(A \wedge B) = \neg A \vee \neg B$ DeMorgan's

Proof:
i) The proof is by case analysis.

Case 1: $A \geq B$
Then $A \vee B = A$ and the negation on the left is $1 - A$. Since $A \geq B$, $1 - A \leq 1 - B$, therefore, the minimum on the right is $1 - A$.

Case 2: $B \geq A$
This proof is identical to that of case 1 with the roles of A and B interchanged.

ii) The proof is by case analysis

Case 1: $A \geq B$
Then $A \wedge B = B$ and the negation on the left is $1 - B$. Since $A \geq B$, $1 - B \geq 1 - A$, therefore the maximum on the right is $1 - B$.

Case 2: $B \geq A$
This proof is identical to that of the case 1 with the roles of A and B interchanged.

Section 2

Other Connectives in Fuzzy Logic

Using the equivalencies that were defined in the chapter on classical logic, it is possible to construct fuzzy logic analogues of the implication, logical equality, exclusive or, joint denial and alternative denial connectives.

Definition 3.2.1: If x and y are variables in a fuzzy logic:

i) $x \rightarrow y = \neg x \vee y = \max\{ 1 - x, y \}$.
ii) $x \leftrightarrow y = (\neg x \vee y) \wedge (\neg y \vee x) = \min\{ \max\{ 1 - x, y \}, \max\{ 1 - y, x \} \}$.
iii) $x \uparrow y = (x \wedge \neg y) \vee (\neg x \wedge y) = \max\{ \min\{ x, 1 - y \}, \min\{ 1 - x, y \} \}$.
iv) $x \downarrow y = \neg x \wedge \neg y = \min\{ 1 - x, 1 - y \}$.
v) $x \mid y = \neg x \vee \neg y = \max\{ 1 - x, 1 - y \}$.

While these formulas provide a way to express these connectives, the value of some of them can be questioned. For example, in classical logic, the $\leftrightarrow$ connective symbolizes

logical equality and ^ logical inequality. When dealing with fuzzy variables, equality of values would be a very rare thing. Furthermore, it is a stretch to argue that min{ max{ 1 – x, y }, max{ 1 – y, x } } is a representation of equality and max{ min{ x. 1 - y }, min{ 1 – x, y } } a representation of inequality. Therefore, if the ↔ or ^ connectives are defined in a fuzzy logic, alternate definitions are generally used.

Section 3
Tautologies and Contradictions In Fuzzy Logic

In fuzzy logic, 1.0 is the equivalent of true and 0.0 the equivalent of false. While it is possible to have expressions that evaluate to these values, in most cases an expression that evaluates to a value close to either 1.0 or 0.0 would be considered significant. In performing a probability analysis of circumstances, there are two different levels of significance that are widely used.

Definition 3.3.1: In the probabilistic analysis of events, if the probability of the events occurring by chance is less than 0.05, then the result is said to be **statistically significant**. If the probability of the events occurring by chance is less than 0.01 then the result is said to be **highly significant**.

Definition 3.3.2: If a fuzzy expression always evaluates to a value greater than 0.95, then the expression is said to be a **significant tautology**. A fuzzy expression that always evaluates to a value greater than 0.99 is said to be a **highly significant tautology**. If an expression always evaluates to a value less than 0.05, then it is said to be a **significant contradiction** and an expression that always evaluates to a value less than 0.01 is said to be a **highly significant contradiction**. If an expression always evaluates to 1.0, it is an **absolute tautology** and if it always evaluates to 0.0 it is an **absolute contradiction**.

Absolute tautologies and absolute contradictions exist in fuzzy logic, as $p \vee 1 = 1$ and $p \wedge 0 = 0$

Section 4
Implementing the Fuzzy Connectives in a Computer Program

It is easy to implement the connectives of fuzzy logic in a computer program. The following is an implementation of fuzzy logic in the programming language Java, an object-oriented language that is widely used to build large programs. Since it is object-oriented, it is based on structures known as **classes**, so we will start with a fuzzy class that will represent the variables in fuzzy logic.

```
/* This class is the definition of the elements of a fuzzy logic. Developed by Charles
Ashbacher, December 30, 2000. */
public class fuzzy
{
// This is the data value of the class.
 private float value;
```

```
// This is the constructor that is called when the new operator is used to create a fuzzy
// object.
 public fuzzy(float invalue)
 {
  /* Do not use the value if it is not within the acceptable range. */
  if((invalue<0.0) || (invalue>1.0))
  {
   System.out.println("The input value is not within the valid range");
   value=0.0f;
  }
  else
  {
   value=invalue;
  }
 }

// This function allows us to modify the value of the fuzzy object.
 public void setvalue(float invalue)
 {
  /* Do not use the value if it is not within the acceptable range. */
  if((invalue<0.0) || (invalue>1.0))
  {
   System.out.println("The input value is not within the valid range");
   return;
  }
  value=invalue;
 }

// This function allows us to obtain the value of the fuzzy object.
 public float getvalue()
 {
  return value;
 }
}
```

This class can then be used in a program that contains functions that compute the values of the fuzzy connectives.

```
/* This program is designed to implement the basic connectives of fuzzy logic. It
was written by Charles Ashbacher 12/30/2000. */

public class Usingfuzzy
{

// This function performs the fuzzy AND operation.
```

```
 public static fuzzy fuzzyand(fuzzy f1, fuzzy f2)
 {
  float x1=f1.getvalue();
  float x2=f2.getvalue();
  float invalue;
  if(x1>=x2)
  {
   invalue=x2;
  }
  else
  {
   invalue=x1;
  }
  fuzzy f3=new fuzzy(invalue);
  return f3;
 }

// This function performs the fuzzy OR operation.
 public static fuzzy fuzzyor(fuzzy f1, fuzzy f2)
 {
  float x1=f1.getvalue();
  float x2=f2.getvalue();
  float invalue;
  if(x1>=x2)
  {
   invalue=x1;
  }
  else
  {
   invalue=x2;
  }
  fuzzy f3=new fuzzy(invalue);
  return f3;
 }

/* Fuzzy implication is defined as max{ 1.0 - f1.value,f2.value}. */

 public static fuzzy fuzzyimplication(fuzzy f1, fuzzy f2)
 {
  float x1=1.0f - f1.getvalue();
  float x2=f2.getvalue();
  float invalue;
  if(x1>=x2)
  {
   invalue=x1;
  }
```

```
 else
 {
  invalue=x2;
 }
 fuzzy f3=new fuzzy(invalue);
 return f3;
}

/* Fuzzy altdenial is defined as max{ 1.0 - f1.value, 1.0 - f2.value }. */

 public static fuzzy fuzzyaltdenial(fuzzy f1, fuzzy f2)
 {
 float x1=1.0f - f1.getvalue();
 float x2=1.0f - f2.getvalue();
 float invalue;
 if(x1>=x2)
 {
  invalue=x1;
 }
 else
 {
  invalue=x2;
 }
 fuzzy f3=new fuzzy(invalue);
 return f3;
}

/* Fuzzy joint denial is defined as min{ 1.0 - f1.value, 1.0 - f2.value }. */

 public static fuzzy fuzzyjointdenial(fuzzy f1, fuzzy f2)
 {
 float x1=1.0f - f1.getvalue();
 float x2=1.0f - f2.getvalue();
 float invalue;
 if(x1>=x2)
 {
  invalue=x2;
 }
 else
 {
  invalue=x1;
 }
 fuzzy f3=new fuzzy(invalue);
 return f3;
}
```

```
/* Fuzzy logical equivalence is defined as
   min{ max{1.0 - f1.value,f2.value}, max{f1.value,1.0 - f2.value} }. */

 public static fuzzy fuzzyequivalence(fuzzy f1, fuzzy f2)
 {
  float f1_value=f1.getvalue();
  float f2_value=f2.getvalue();
  float OneMinusf1_value=1.0f - f1_value;
  float OneMinusf2_value=1.0f - f2_value;
  float invalue, firstentry,secondentry;
  if(OneMinusf1_value>=f2_value)
  {
   firstentry=OneMinusf1_value;
  }
  else
  {
   firstentry=f2_value;
  }
  if(OneMinusf2_value>=f1_value)
  {
   secondentry=OneMinusf2_value;
  }
  else
  {
   secondentry=f1_value;
  }
  if(firstentry>=secondentry)
  {
   invalue=secondentry;
  }
  else
  {
   invalue=firstentry;
  }
  fuzzy f3=new fuzzy(invalue);
  return f3;
 }

/* Fuzzy logical exclusive or is defined as
   max{ min{1.0 - f1.value,f2.value}, min{f1.value,1.0 - f2.value} }. */

 public static fuzzy fuzzyxor(fuzzy f1, fuzzy f2)
 {
  float f1_value=f1.getvalue();
  float f2_value=f2.getvalue();
  float OneMinusf1_value=1.0f - f1_value;
```

```
 float OneMinusf2_value=1.0f - f2_value;
 float invalue, firstentry,secondentry;
 if(OneMinusf1_value>=f2_value)
 {
  firstentry=f2_value;
 }
 else
 {
  firstentry=OneMinusf1_value;
 }
 if(OneMinusf2_value>=f1_value)
 {
   secondentry=f1_value;
 }
 else
 {
   secondentry=OneMinusf2_value;
 }
 if(firstentry>=secondentry)
 {
   invalue=firstentry;
 }
 else
 {
   invalue=secondentry;
 }
 fuzzy f3=new fuzzy(invalue);
  return f3;
}

// This function performs fuzzy negation.
 public static fuzzy fuzzynegate(fuzzy f1)
 {
  float x1=f1.getvalue();
  fuzzy f3=new fuzzy(1.0f-x1);
  return f3;
 }

// This is the function where execution begins. The values passed into the fuzzy objects
// are the values that they will have. All three of the functions matching the connectives
// are used and the output is just to demonstrate them in action.
 public static void main(String args[])
 {
  fuzzy f1=new fuzzy(0.65f);
  fuzzy f2=new fuzzy(0.32f);
  fuzzy f3;
```

```
 f3=fuzzynegate(f1);
 System.out.println("The value of f3 after fuzzy negation is "+f3.getvalue());
 f3=fuzzyand(f1,f2);
 System.out.println("The value of f3 after fuzzy ANDing is "+f3.getvalue());
 f3=fuzzyor(f1,f2);
 System.out.println("The value of f3 after fuzzy ORing is "+f3.getvalue());
f3=fuzzyimplication(f1,f2);
 System.out.println("The value of f3 after fuzzy implication is "+f3.getvalue());
 f3=fuzzyequivalence(f1,f2);
 System.out.println("The value of f3 after fuzzy equivalence is "+f3.getvalue());
 f3=fuzzyxor(f1,f2);
 System.out.println("The value of f3 after fuzzy exclusive OR is "+f3.getvalue());
 f3=fuzzyaltdenial(f1,f2);
 System.out.println("The value of f3 after fuzzy alternate denial is "+f3.getvalue());
 f3=fuzzyjointdenial(f1,f2);
 System.out.println("The value of f3 after fuzzy joint denial is "+f3.getvalue());
 }
}
```

When this program is run, the output is

```
The value of f3 after fuzzy negation is 0.35000002
The value of f3 after fuzzy ANDing is 0.32
The value of f3 after fuzzy ORing is 0.65
The value of f3 after fuzzy implication is 0.35000002
The value of f3 after fuzzy equivalence is 0.35000002
The value of f3 after fuzzy exclusive OR is 0.65
The value of f3 after fuzzy alternate denial is 0.68
The value of f3 after fuzzy joint denial is 0.35000002
```

Section 5

Rules of Inference in Fuzzy Logic

The rules of inference in fuzzy logic are very similar to those in three-valued logic. However, there is a key difference in that a numeric value can be assigned to some of the results.

Definition 3.5.1: A rule of inference

$$\frac{p_1, p_2, \ldots, p_k}{q}$$

is **z-valid** in the fuzzy logic if whenever each of the expressions $p_1, p_2, \ldots, p_k$ is greater than or equal to z, the expression q has a value that is greater than or equal to z. If the only conclusion is that q is greater than zero, then it is **nonzero valid**.

Given the definition of the $\rightarrow$ connective for fuzzy logic, the modus ponens inference rule does not apply. It is possible for $p \rightarrow q$ and p to be nonzero while q is zero. As was demonstrated in chapter 2, that is not necessarily the last word on modus ponens. It is also possible to define the $\rightarrow$ connective so that it is $\neg p \vee q$ if $p \leq q$ and 0 if $p > q$. If this definition is used, then if $p \rightarrow q$ and p are nonzero, it follows that q is also nonzero.

If it is properly qualified, the and-elimination rule of inference also can be used in fuzzy logic

Given $p_1 \wedge p_2 \wedge \ldots \wedge p_k = v$, then p_I can be inferred to have a value $\geq v$.

The and-introduction rule of inference also has a qualified meaning in fuzzy logic.

Given that $p_1, p_2, \ldots, p_k$ are all nonzero and greater than or equal to v, we can infer that $p_1 \wedge p_2 \wedge \ldots \wedge p_k$ is also greater than or equal to v.

The or-introduction rule of inference has a qualified meaning in fuzzy logic.

Given p, we can infer $p \vee q$ having a value greater than or equal to p.

The unit-resolution rule of inference does not apply in fuzzy logic, for if $p \vee q$ and $\neg q$ are nonzero, then it is not possible to conclude that p is nonzero. Use the values of p = 0 and q = ½ to verify this.

The modus tollens rule of inference also does not apply in fuzzy logic if the $\rightarrow$ connective is defined as $\neg p \vee q$. However, if the definition is $\neg p \vee q$ if $p \leq q$ and 0 if $p > q$, then if $p \rightarrow q$ and $\neg q$ are nonzero, then it follows that p is also nonzero.

The resolution rule of inference also applies in fuzzy logic. For if, $p \rightarrow q$ and $q \rightarrow r$ are nonzero, then $p \rightarrow r$ is also nonzero. This can be verified by the examination of the truth table for the three variables p, q and r.

The definition of formal reasoning in fuzzy logic is similar to the definitions of the inference rules.

Definition 3.5.2: Let $\Gamma = \{ a_1, a_2, \ldots, a_k \}$ represent a set of wffs in fuzzy logic. A wff p is said to be a **consequence of Γ** if whenever all the elements of Γ are nonzero, then p is also nonzero.

Definition 3.5.3: Let $\Gamma = \{ a_1, a_2, \ldots, a_k \}$ represent a set of wffs in fuzzy logic. A wff p is said to be an i**ncreased consequence of Γ** if p is a consequence of Γ and the value of p is greater than or equal to all the elements of Γ.

The notation for a consequence in fuzzy logic is the same as that for classical logic.

$$a_1, a_2, \ldots, a_k \vdash p.$$

The notation for an increased consequence in fuzzy logic is

$a_1, a_2, \ldots, a_k \vdash_* p.$

Examples:
Given the rules of inference in fuzzy logic, the following are all valid inferences.

$p,q \vdash p \wedge q,$

$p \wedge q \vdash p, \quad p \wedge q \vdash_* p$

$p \vdash p \vee q$

Section 6
Modal Logic With Fuzzy Variables

Modal logic is a form of logic where the notions of necessity and possibility are included. A necessary truth is one that always holds and a possible truth is one that can be true. The difference is that now there is the inclusion of different possible worlds. A world is a set of parameters describing the circumstances of the logical variables and the definitions of the parameters are called an **interpretation**. Therefore, a necessary truth is one that is true for all possible interpretations and a contingent truth is one that is true in at least one interpretation.

Example:
Let the set of discourse be the integers greater than zero. Suppose we define the operation of "addition plus k (+k)" to be the infinite set of operations

$$m +k\ n = m + n + k \text{ for m and n integers and } k \geq 1.$$

In other words, the operation addition plus k is the sum of the two integers plus k.

The statement

Addition plus k is always commutative.

is a necessary truth, since

$$m +k\ n = m + n + k = n + m + k = n +k\ m \text{ for all values of k.}$$

However, the statement

m +k 0 = m
is a contingent truth, as it is only true when k = 0.

Definition 3.6.1: If A is an expression, then **LA** is read A is a necessary truth and **MA** is read A is a contingent truth. Clearly, for any expression A, if LA then MA.

Variables in a fuzzy logic can be assigned different values based on the different circumstances, where the different sets of circumstances make up a world.

Definition 3.6.2: A **modal frame** in a fuzzy logic is a structure with the following characteristics:

i) A set of constants 0 and 1, where 0 is always 0.0 and 1 is 1.0.
ii) A set of variables V = { x, y, z, . . . } that can have the values from 0.0 to 1.0 inclusive. This set may be finite or infinite.
iii) W = {w_1, w_2, . . . }is a non-empty set of possible worlds or interpretations of the variables and operators. This set may also be finite or infinite.
iv) A set of functions that accepts an element of V (the variable)and an element of W (the world) and assigns a number between 0.0 and 1.0 inclusive. This function assigns values to the variables in this world.
v) The set of logical operators that can be used on the variables.
vi) A set of relations that accepts an element of W and an element of the set of logical operators and assigns a function that defines the interpretation of that operator in that world.
vii) A set of rules that defines the set of wffs in the modal frame.

Example:
The major league baseball teams in the United States are split up into two leagues, the National and American. At the end of the season, one team from each league meets in the World Series to decide the championship.

The following is the definition of a modal frame in fuzzy logic.

The set of variables is V = { Cubs, Braves, . . ., Yankees }, one for each major league baseball team. Each variable is assigned a value that represents the likelihood that it will appear in the world series.

The set of logical operators is { /\, \/, ¬ }
The set of wffs is defined in the following way

i) Every element of V or { 0,1 } is a wff.
ii) If A and B are wffs, then (A), A \/ B, A /\ B and ¬A are wffs.
iii) Only expressions that can be formed using (i) and (ii) are wffs.

There is only one world and it is the set of teams in major league baseball.

If x is an element of V, then ¬x = 1 – x.
If x and y are elements of V, then x ∧ y = 0.0 if the teams are in the same league and min { x, y} if x and y are teams in different leagues.

If x and y are elements of V, then x ∨ y = x + y if the teams are in the same league and max {x, y} if the teams are in different leagues.

One example of a function that assigns values to the variables in this world is as follows:

National League		American League	
Team	Value	Team	Value
Cubs	0.06	Yankees	0.16
Braves	0.09	Red Sox	0.10
Expos	0.04	Orioles	0.05
Mets	0.09	Devil Rays	0.02
Marlins	0.03	Blue Jays	0.03
Phillies	0.07	White Sox	0.09
Reds	0.06	Twins	0.06
Pirates	0.05	Indians	0.13
Cardinals	0.09	Tigers	0.02
Astros	0.10	Royals	0.03
Brewers	0.02	Mariners	0.12
DiamondBacks	0.12	Athletics	0.07
Dodgers	0.05	Angels	0.07
Giants	0.09	Rangers	0.05
Padres	0.06		
Rockies	0.04		

Where the sums of the values is 1.0 for each league. This is consistent with the rules of probability, where a certain event has probability one.

Section 7
Temporal Logic

Clearly, as the baseball season progresses, the probabilities assigned to the teams will alter based on their performance. **Temporal logic** is logic where the values can change over time. Altering the modal logic in the previous section so that it is temporal requires only a modification of the function that assigns the values to the variables in the world.

iv) A set of functions that accepts an element of V (the variable), an element of W (the world) and a time parameter t and assigns a number between 0.0 and 1.0 inclusive. This function assigns values to the variables in this world.

For the previous example, the function that assigns the probabilities would have to have a label that lists the time when the probabilities were assigned. Additional operators used in temporal logic reflect the passage of time.

Definition 3.7.1: Let A be an expression in a temporal logic.

i) FA – A will be true at some future time.
ii) PA – A was true at some past time.
iii) GA – A will be true at all future times.
iv) HA – A has always been true in the past.

Chapter 4

Neutrosophic Logic

Section 1

Definition of Neutrosophic Logic

Neutrosophic logic was created by Florentin Smarandache (1995) and is an extension/combination of the fuzzy logic, intuitionistic logic, paraconsistent logic, and the three-valued logics that use an indeterminate value. In neutrosophic logic, in an easy way, every logical variable x is described by an ordered triple.

$$x= (t, i, f)$$

where t is the degree of truth, f is the degree of false and i is the level of indeterminacy.

A) To maintain consistency with the classical and fuzzy logics and with probability, there is the special case where $t + i + f = 1$.

B) But to refer to intuitionistic logic, which means incomplete information on a variable, proposition or event one has $t + i + f < 1$.

C) Analogically, referring to paraconsistent logic, which means contradictory sources of information about a same logical variable, proposition, or event one has $t + i + f > 1$.

Note: An alternate definition used by Smarandache is to have the three values of the ordered triple integers greater than or equal to zero and $t + i + f <$ or = or > 100. This definition is of course consistent with percentages. For our purposes, the use of real numbers that sum to 1.00 is preferable, as it is more consistent with other logics.

A more **general definition** later developed by Smarandache (1999-2002)[1] is:
"Let T, I, F be standard or non-standard real subsets of the non-standard unit interval $]^-0, 1^+[$[2],

with sup T = t_sup, inf T = t_inf,
sup I = i_sup, inf I = i_inf,
sup F = f_sup, inf F = f_inf,

and n_sup = t_sup+i_sup+f_sup,
n_inf = t_inf+i_inf+f_inf.

---

[1] Smarandache, Florentin (1999). *A Unifying Field in Logics: Neutrosophic Logic. Neutrosophy, Neutroosphic Set, Neutrosophic Probability and Statistics*, American Research Press, Rehoboth; http://www.gallup.unm.edu/~smarandache/eBook-Neutrosophics2.pdf
We reproduce, with publisher's permission, the neutrosophic logic definition's paragraph below.

[2] There are two notations used by different authors (F. Smarandache, J. Dezert, Charles T. Le, A. Buller, J. Allen, M. Khoshnevisan, S. Singh, S. Bhattacharya, F. Liu, Gh. C. Dinulescu-Campina, C. Lucas, C. Gershenson) for the non-standard unit interval: $]^-0, 1^+[$ and $\shortparallel^-0, 1^+\shortparallel$. Both of them are okay.

The sets T, I, F are not necessarily intervals, but may be any real sub-unitary subsets: discrete or continuous; single-element, finite, or (countable or uncountable) infinite; union or intersection of various subsets; etc.

They may also overlap. The real subsets could represent the relative errors in determining t, i, f (in the case when the subsets T, I, F are reduced to points).
Statically T, I, F are subsets.

But dynamically, looking therefore from another perspective, the components T, I, F are at each instance dependant on many parameters, and therefore they can be considered set-valued vector functions or even operators.

The parameters can be: time, space, etc. (some of them are hidden/unknown parameters): T(t, s, …), I(t, s, …), F(t, s, …), where t=time, s=space, etc., which allows the neutrosophic logic to be used in quantum physics. The Dynamic Neutrosophic Calculus can be also used in psychology and the Neutrosophics try to reflect the dynamics of things and ideas.

For example:
The proposition "Tomorrow it will be raining" does not mean a fixed-valued components structure; this proposition may be say 40% true, 50% indeterminate, and 45% false at time t_1; but at time t_2 may change at 50% true, 49% indeterminate, and 30% false (according with new evidences, sources, etc.); and tomorrow at say time t_{145} the same proposition may be 100%, 0% indeterminate, and 0% false (if tomorrow it will indeed rain). This is the dynamics: the truth value changes from one time to another.

In other examples: the truth value of a proposition may change from a place to another place, for example: the proposition “It is raining” is 0% true, 0% indeterminate, and 100% false in Albuquerque (New Mexico), but moving to Las Cruces (New Mexico) the truth value changes and it may be (1, 0, 0).

Also, the truth value depends/changes with respect to the observer (subjectivity is another parameter of the functions/operators T, I, F). For example: “John is smart” can be (.35, .67, .60) according to his boss, but (.80, .25, .10) according to himself, or (.50, .20, .30) according to his secretary, etc.

T, I, and F are called *neutrosophic components*, representing the truth, indeterminacy, and falsehood values respectively referring to neutrosophy, neutrosophic logic, neutrosophic set, neutrosophic probability, neutrosophic statistics.

This representation is closer to the reasoning of the human mind. It characterizes/catches the imprecision of knowledge or linguistic inexactitude perceived by various observers (that’s why T, I, F are subsets - not necessarily single-elements), uncertainty due to incomplete knowledge or acquisition errors or stochasticity (that’s why the subset I exists),

and vagueness due to lack of clear contours or limits (that's why T, I, F are subsets and I exists; in particular for the appurtenance to the neutrosophic sets).

One has to specify the superior (x_sup) and inferior (x_inf) limits of the subsets because in many problems arises the necessity to compute them.

Definition of Neutrosophic Logic[3]:
A logic in which each proposition is estimated to have the percentage of truth in a subset T, the percentage of indeterminacy in a subset I, and the percentage of falsity in a subset F, where T, I, F are defined above, is called *Neutrosophic Logic*.

We use a subset of truth (or indeterminacy, or falsity), instead of a number only, because in many cases we are not able to exactly determine the percentages of truth and of falsity but to approximate them. For example a proposition is between 30-40% true and between 60-70% false, even worse: between 30-40% or 45-50% true (according to various analyzers), and 60% or between 66-70% false.

The subsets are not necessary intervals, but any sets (discrete, continuous, open or closed or half-open/half-closed interval, intersections or unions of the previous sets, etc.) in accordance with the given proposition. A subset may have one element only in special cases of this logic.

Constants: (T, I, F) truth-values, where T, I, F are standard or non-standard subsets of the non-standard interval $]^-0, 1^+[$, where $n_{inf} = \inf T + \inf I + \inf F \geq {}^-0$, and $n_{sup} = \sup T + \sup I + \sup F \leq 3^+$.
Atomic formulas: a, b, c, … .
Arbitrary formulas: A, B, C, … .

The neutrosophic logic is a formal frame trying to measure the truth, indeterminacy, and falsehood." (F. Smarandache)

Smarandache's hypothesis is that **no theory is exempted from paradoxes**, because of the imprecision of the language, metaphoric expression, various levels or meta-levels of understanding/interpretation which might overlap.

The advantage of using neutrosophic logic is that this logic distinguishes between relative truth, that is a truth in one or a few worlds only, noted by 1, and absolute truth, that is a truth in all possible worlds, noted by 1^+. And similarly, neutrosophic logic distinguishes between relative falsehood, noted by 0, and absolute falsehood, noted by $^-0$.

In neutrosophic logic the sum of components is not necessarily 1 as in classical and fuzzy logic, but any number between $^-0$ and 3^+, and this allows the neutrosophic logic to be able to deal with paradoxes, propositions which are true and false in the same time: thus NL(paradox)=(1, I, 1); fuzzy logic can not do this because in fuzzy logic the sum of components should be 1.

[3] Also called **Smarandache Logic** in the Dictionary of Computing, by Dennis Howe, England.

Remarks: In this book we study only the special case where the components T, I, F are subsets reduced to one single element each, respectively to t, i, and f; also one ignores the distinction between relative and absolute truth/falsehood/indeterminacy. And then this case is divided into three subcases:

a) When the sum of components t + i + f = 1 (classical and fuzzy logic);
b) When the sum of components is t + i + f < 1 (intuitionistic logic);
c) When the sum of components is t + i + f ≥ 1 (paraconsistent logic).

Definition 4.1.1: An element of an **Intuitionistic Neutrosophic Logic (INL)** is a four-tuple
(t, i, f, u) where t + i + f + u = 1.0 and u ≥ 0.0. t is the degree of truth, i the value of indeterminancy, f the degree of falsehood and u is the degree to which the circumstances are unknown.

Definition 4.1.2: An element of a **Paraconsistent Neutrosophic Logic (PNL)** is a three-tuple
(t, i, f) where t + i + f ≥ 1.0.

Note: It is possible to define a general form of logic as a set of four-tuples where the values are allowed to vary over greater ranges. However, for the purposes of simplifying automated reasoning in Neutrosophic logic, the three separate definitions are used.

In our next book dedicated to neutrosophic logic we'll be studying the general case as well (when T, I, F are subsets in the non-standard analysis of $]^{-}0, 1^{+}[$, and also we'll make distinction between relative {1 or 0} and absolute truth/falsehood/indeterminacy $\{^{+}1$ or $^{-}0\}$). While the sum of inferior/superior components satisfies the below inequality: $^{-}0 \leq \inf T+\inf I+\inf F \leq \sup T+\sup I+\sup F \leq 3^{+}$.

Example:
If i is always zero and t and f must be zero or one, then the variables are restricted to the forms (1,0,0) and (0,0,1). The behavior of the connectives in CS for these values can be defined using the following truth tables.

p	Q	p ∧ q	p ∨ q	¬p
(1,0,0)	(1,0,0)	(1,0,0)	(1,0,0)	(0,0,1)
(1,0,0)	(0,0,1)	(0,0,1)	(1,0,0)	(0,0,1)
(0,0,1)	(1,0,0)	(0,0,1)	(1,0,0)	(1,0,0)
(0,0,1)	(0,0,1)	(0,0,1)	(0,0,1)	(1,0,0)

If we adopt the abbreviations T = (1,0,0) and F = (0,0,1), then it is clear that this model is equivalent to classical logic.

Example:
If the values of the variables in NL are restricted to (1, 0, 0), (0, 0, 1) and (½, 0, ½) and the behavior of the connectives defined as they were for the Lukasiewicz three-valued logic, then the system is isomorphic to the Lukasiewicz three-valued logic.

Example:
If i is always zero and t and f have only the restrictions of being between zero and one, then the ¬, ∧ and ∨ connectives can be defined in the following way:

$\neg(t,0,f) = (f,0,t)$ $\quad (t_1,0,f_1) \wedge (t_2,0,f_2) = (\min\{ t_1,t_2\}, 0, \max\{ f_1,f_2 \})$

$(t_1,0,f_1) \vee (t_2,0,f_2) = (\max\{ t_1,t_2\}, 0, \min\{ f_1,f_2 \})$.

Theorem 4.1.1: If x is a variable in a fuzzy logic, then the mapping $C: x \leftrightarrow (x, 0, 1\text{-}x)$ using the logical operators as defined in the previous example is an isomorphism between the fuzzy logic and a subclass of neutrosophic logic where i is always zero and the three elements sum to 1.

Proof: If x and y are variables in a fuzzy logic, then they would be mapped to $(x, 0, 1 – x)$ and $(y, 0, 1 – y)$ respectively.

$C(\neg x) = C(1 – x) = (1 – x, 0, x)$ and $\neg C(x) = \neg (x, 0, 1 – x) = (1 – x, 0, x)$, therefore

$C(\neg x) = \neg C(x)$.

$C(x \wedge y) = C(\min \{x, y\}) = (\min \{x,y\}, 0, 1 – \min \{x,y\})$

$C(x) \wedge C(y) = (x, 0, 1 – x) \wedge (y, 0, 1 – y) = (\min \{ x,y\}, 0, \max \{ 1 – x, 1 – y \})$

Given that $1 = \min \{x,y\} + \max \{ 1 – x, 1 – y \} = \min \{x,y\} + (1 – \min\{x,y\})$ it follows that

$\max \{ 1 – x, 1 – y \} = (1 – \min\{x,y\})$ and $C(x \wedge y) = C(x) \wedge C(y)$.

$C(x \vee y) = C(\max \{x, y\}) = (\max \{x,y\}, 0, 1 – \max \{x,y\})$

$C(x) \vee C(y) = (x, 0, 1\text{-}x) \vee (y, 0, 1 – y) = (\max\{x,y\}, 0, \min\{x,y\})$

Once again, given that $1 = \max\{x,y\} + (1 – \max\{x,y\}) = \max\{x,y\} + \min\{x,y\}$, it follows that

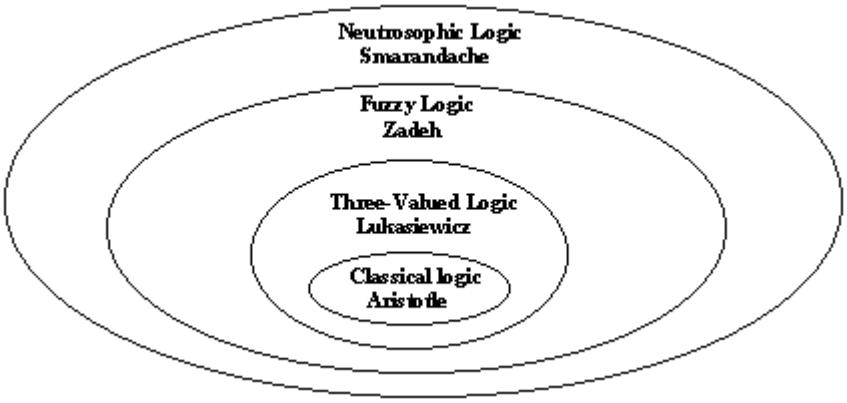
$1 – \max\{x,y\} = \min\{x,y\}$ amd $C(x \vee y) = C(x) \vee C(y)$.

Therefore, the mapping C is a homomorphism.

If $(x , 0, 1 – x) = (x_1, 0, 1 – x_1)$, then by the definition of an ordered triple, $x = x_1$. Therefore, C is a 1-1 mapping and is an isomorphism.

From the previous examples, it should be clear that Neutrosophic Logic is a generalization of the classical, three-valued and fuzzy logics. The progression of these logics is summarized in figure 1.

Figure 1



In fact, neutrosophic logic can be considered a more accurate representation of the world we are in. Few things are absolutes, in fact mathematics is one of the few areas where something is known with certainty. The indeterminate entry allows for the acknowledgement that the values given for the true and false entries are commonly not known with certitude.

There are other, similar definitions of the variables in NL that can more accurately represent some more specialized circumstances. Consider the taking of a statistical sampling such as a public opinion poll where two options are possible. As a consequence of the laws of statistics, such surveys always have a sampling error. The results of such samplings are generally presented in the form:

x% first answer, y% second answer with a margin of error ±k%.

Generally, $x + y = 100$. What this means is that the first answer could be as high as $x+k\%$ where the second answer would have a corresponding value of $y – k\%$ to the first answer being as low as $x – k\%$ with the second being $y + k\%$.

Example: A group of people are surveyed as to their preference for candidate A versus candidate B. Suppose that the answers are 55% favor candidate A, and 45% candidate B

and the survey has an error of plus or minus 5 percent. This means that the true values are somewhere between 60% for A and 40% for B to 50% for A and 50% for B.

Such situations can be handled with the following modification of the definition of the items in a NL.

Definition 4.1.3: A Neutrosophic Logic where indeterminacy is an error range (**NLE**) is a triple (t, i, f) where $t + f = 1.0$, $i \leq \min\{ t,f \}$, $t + i \leq 1.0$ and $f + i \leq 1.0$. It is interpreted to mean that the range of values for the truth can be anywhere in the range $[t – i, t + i]$ where the corresponding values for the false component would be $[f + i$ to $f – i]$. The limitations on the values are necessary so that there is no probability that is either less than zero or greater than one.

Thus one can get $t+i+f>1$.

Examples:
If a poll is taken and the results are

Forty-five percent of the people surveyed approve of the job the president is doing with an error of plus or minus 5%.

The corresponding NLE expression would be (.45,.05,.55) which means that this survey indicates that the actual percentage of people with a favorable approval rating is somewhere in the range 0.40 to 0.50.

If the values of the variables are allowed to change according to circumstances, then the triplets of NL can be used to represent additional circumstances. For example, if we are given an expression of the form

$$x > 0$$

then it is not possible to assign it a truth value because of the unknown value of x.

However, the value (0, 1, 0) can be assigned in NL, as the truth value is completely indeterminate. However, once x is assigned a value, then the value will be either (1,0,0) or (0,0,1). Furthermore, if we know something about the universal set of possible values for x, then values can be assigned to the t and f portions. For example, if the set of discourse is all real numbers and the choice of x is random, we could argue that the value is (0.50,0.0,0.50). This could be interpreted that the odds are fifty-fifty that x is larger than zero.

This situation occurs in computer science. If a variable is declared without being assigned a value,

```
int m;
```

then the contents of the variable are considered garbage, meaning that the value is just whatever is leftover from previous operations. The current status of the variable could be described by the value (0.0,1.0,0.0).

In quantum mechanics it is possible for tiny particles such as electrons to possess one of two possible spin states, "up" and "down." They are placed in quotes because the spin property of an electron is different from that we are familiar with. However, until the spin property of the electron is examined, it is in a state of uncertainty, being considered either half up and half down or indeterminate. If we use the half-and-half interpretation, then the NL representation would be (½, 0, ½). However, if we consider it indeterminate, then the NL representation would be (0, 1, 0). Once the spin state of the electron is examined, then the representation would be either (1, 0, 0) or (0, 0, 1) where one is mapped to an "up" spin and the other to a "down" spin.

Section 2

Logical Connectives in Neutrosophic Logic

Definition 4.2.1: Let (t_1, i_1, f_1) and (t_2, i_2, f_2) be elements of NL where the sum of the elements of the triplet is 1.0. The logical connectives of { ¬, ∧, ∨ } can be defined in the following way:

$\neg(t_1, i_1, f_1) = (f_1, i_1, t_1)$
$(t_1 , i_1, f_1) \wedge (t_2, i_2, f_2) = (t = \min \{ t_1, t_2\},\ i = 1 - (t + f),\ f = \max \{ f_1, f_2 \})$
$(t_1,\ i_1,\ f_1) \vee (t_2, i_2, f_2) = (t = \max \{ t_1, t_2\},\ 1 - (t + f), f = \min \{ f_1, f_2 \}).$

Since there are other ways to define the connectives, so we will note this set as NL1.

These definitions are consistent with our common notions of indeterminacy. If an event is assigned an indeterminate value of i, then the negation cannot have an indeterminacy value that is different. If it was less indeterminate, then the t and f values would have to have higher values and if it was more indeterminate, then the values of t and f would have to be smaller. The swapping of the t and f values is consistent with how negation is defined in other logical structures.

If we have two triples in NL1, then the value of t for an and operation must be known to at least the smallest of the two t values. The value of f for an and operation must be known to the maximum of the two f values. For the or operation the value of t must be known to at least the maximum of the two t values and the value of f must be known to the minimum of the f values.

This interpretation of the ∧ and ∨ connectives is consistent with how true and false are interpreted in the classical, Lukasiewicz three-valued and fuzzy logics. With the ∧ connective, the false value dominates and with the ∨ connective the true dominates.

An alternate definition of the ∨, ∧ and ¬ connectives that uses the indeterminate as the dominant value is consistent with the dominance of the M or meaningless value in Bochvar's logic.

Definition 4.2.2: Let (t_1, i_1, f_1) and (t_2, i_2, f_2) be elements of NL. The logical connectives of
$\{ \neg, \wedge, \vee \}$ can be defined in the following way:

$$\neg(t_1, i_1, f_1) = (f_1, i_1, t_1)$$

$$(t_1 , i_1, f_1) \wedge (t_2, i_2, f_2) = (1 - \max \{i_1,i_2 \} - \min \{ 1 - \max \{i_1,i_2 \}, \max \{ f_1, f_2 \}\}, \max \{i_1,i_2 \}, \min \{ 1 - \max \{i_1,i_2 \}, \max \{ f, f_1 \}\})$$

In other words, for the conjunction, the indeterminate value dominates first, followed by the false.

$$(t_1, i_1, f_1) \vee (t_2, i_2, f_2) = (\min \{ 1 - \max \{i_1,i_2 \}, \max \{ t_1, t_2 \}\}, \max \{i_1,i_2 \},$$

$$1 - \max \{i_1,i_2 \} - \min \{ 1 - \max \{i_1,i_2 \}, \max \{ t_1, t_2 \}\}).$$

In other words, for the disjunction, the indeterminate value dominates first, followed by the true.

The elements of NL with these definitions of the connectives will be referred to by NL2.

Definition 4.2.3: Let (t_1, i_1, f_1, u_1) and (t_2, i_2, f_2, u_2) be elements of INL. The logical connectives of $\{ \neg, \wedge, \vee \}$ can be defined in the following way:

$$\neg(t_1, i_1, f_1, u_1) = (f_1, i_1, t_1, u_1)$$

$$(t_1 , i_1, f_1, u_1) \wedge (t_2, i_2, f_2, u_2) = (t = \min \{ t_1, t_2\}, i = \min\{ i_1, i_2 \}, f = \max \{ f_1, f_2 \}, u = 1 - t - i - f)$$

$$(t_1, i_1, f_1, u_1) \vee (t_2, i_2, f_2, u_2) = (t = \max \{ t_1, t_2\}, i = \min\{ i_1, i_2 \}, f = \min \{ f_1, f_2 \}, u = 1 - t - i - f).$$

The definition of the negation is consistent with the negation in other forms of logic, in that it is simply a reversal of the values of the true and false components. It also makes sense that the indeterminate and unknown elements are not altered by the taking of a negation.

This definition is also consistent with the domination of true and false in classical logic, in that false dominates in a conjunction and true dominates in a disjunction. With this definition, the indeterminate values are reduced and the unknown values increased.

The elements of INL with these definitions of the connectives will be referred to by INL1.

Definition 4.2.4: Let (t_1, i_1, f_1, u_1) and (t_2, i_2, f_2, u_2) be elements of INL. The logical connectives of $\{ \neg, \wedge, \vee \}$ can be defined in the following way:

$\neg(t_1, i_1, f_1, u_1) = (f_1, i_1, t_1, u_1)$

$(t_1 , i_1, f_1, u_1) \wedge (t_2, i_2, f_2, u_2) = (t = \min \{ t_1, t_2\},\ i = 1 – t – f - u, f = \max \{ f_1, f_2 \},$
$u = \min \{ u_1, u_2 \})$

$(t_1,\ i_1,\ f_1, u_1) \vee (t_2, i_2, f_2, u_2) = (t = \max \{ t_1, t_2\}, i = 1 – t – f - u,$
$f = \min \{ f_1, f_2 \}, u = \min \{ u_1, u_2 \}).$

This definition also maintains the same consistency of the negation with other forms of logic. True still dominates in a disjunction and false in a conjunction. With this definition, the unknown values tend to be reduced with a corresponding increase in the indeterminate value.

The elements of INL with these definitions of the connectives will be referred to by INL2.

Definition 4.2.5: Let (t_1, i_1, f_1) and (t_2, i_2, f_2) be elements of PNL. The logical connectives of
$\{ \neg, \wedge, \vee \}$ can be defined in the following way:

$\neg(t_1, i_1, f_1) = (f_1, i_1, t_1)$

$(t_1 , i_1, f_1) \wedge (t_2, i_2, f_2) = (t = \min \{ t_1, t_2\},\ i = \max \{ i_1, i_2 \},\ f = \max \{ f_1, f_2 \})$

$(t_1,\ i_1,\ f_1) \vee (t_2, i_2, f_2) = (t = \max \{ t_1, t_2\}, i = \max \{ i_1, i_2 \}, f = \min \{ f_1, f_2 \}).$

Once again, the definition of negation is consistent with that of other logics. The principle of the domination of false in conjunction and true in disjunction is also maintained, with the primary difference from other logics being the simultaneous domination of the indeterminate value.

The elements of PNL with these definitions of the connectives will be referred to by PNL1.

Theorem 4.2.1: If (t_1, i_1, f_1) and (t_2, i_2, f_2) are elements in NL1 , then

(i) $\neg(t_1, i_1, f_1)$

(ii) $(t_1, i_1, f_1) \wedge (t_2, i_2, f_2)$

(iii) $(t_1, i_1, f_1) \vee (t_2, i_2, f_2)$

are also elements in NL1. Therefore, NL1 is closed under these definitions of the logical operations.

Proof:

(i) If $t_1 + i_1 + f_1 = 1.00$, then $f_1 + i_1 + t_1$ is also 1.00.

(ii) By the choice of the indeterminate term, it is only necessary to prove that $\min \{ t_1, t_2 \} + \max \{ f_1, f_2 \} \leq 1$. Assume that $(\min \{ t_1, t_2 \} + \max \{ f_1, f_2 \}) > 1$. The min and max values cannot be from the same triple, for if they were, the sum of the values for that triple would have been greater than 1.0. Therefore, the min is from one triple and the max is from the other. Without loss of generality, assume that $\min\{ t_1, t_2 \} = t_1$ and $\max\{ f_1, f_2 \} = f_2$. Since $t_1 + f_2 > 1.00$, it follows that $t_2 < t_1$. However, this contradicts the result that $t_1 \leq t_2$ and means that the assumption $(\min \{ t_1, t_2 \} + \max \{ f_1, f_2 \}) > 1$ is incorrect. Therefore, $(\min \{ t_1, t_2 \} + \max \{ f_1, f_2 \}) \leq 1$ and $\wedge$ is closed.

(iii) By the choice of the indeterminate term, it is only necessary to prove that $\max \{ t_1, t_2 \} + \min \{ f_1, f_2 \} \leq 1$. The proof is similar to that for (ii) and is omitted.

Theorem 4.2.2: If (t_1, i_1, f_1) and (t_2, i_2, f_2) are elements in NL2 , then

(i) $\neg(t_1, i_1, f_1)$

(ii) $(t_1, i_1, f_1) \wedge (t_2, i_2, f_2)$

(iii) $(t_1, i_1, f_1) \vee (t_2, i_2, f_2)$

are also elements in NL2. Therefore, NL2 is closed under these definitions of the logical operations.

Proof: Similar to that of theorem 4.2.1.

Theorem 4.2.3:

i) INL1 is closed under the connectives { $\neg, \wedge, \vee$ }.
ii) INL2 is closed under the connectives { $\neg, \wedge, \vee$ }.
iii) PNL1 is closed under the connectives { $\neg, \wedge, \vee$ }.

Proof:

(i) Let (t_1, i_1, f_1, u_1) and (t_2, i_2, f_2, u_2) be elements of INL1. If $t_1 + i_1 + f_1 + u_1 = 1.0$, then the altering of the order will have no affect. Therefore, INL1 is closed with respect to $\neg$. Since $\min\{t_1, t_2 \} + \min\{i_1, i_2 \} + \max \{f_1, f_2 \} \leq 1.0$, it follows from the choice of u that $t + i + f + u = 1.0$ and INL1 is closed with respect to $\wedge$. A similar argument can be used to verify that INL1 is closed with respect to $\vee$.

(ii) Let (t_1, i_1, f_1, u_1) and (t_2, i_2, f_2, u_2) be elements of INL2. The proof of the closure of $\neg$ is the same as that for INL1. The proofs for the closure of $\wedge$ and $\vee$ are similar to that for INL1, simply interchange the roles of i and u.

(iii) If $t_1 + i_1 + f_1 \geq 1.0$ and $t_2 + i_2 + f_2 \geq 1.0$, then choosing the maximum of two of the three positions must lead to a sum that is also greater than or equal to 1.0.

Definition 4.2.6: Given A and B two expressions in NL_1, we use $A =_{NL1} B$ to mean that A and B always have the same triplet values for the same assignments to their common components. The singleton equality symbol = will be used to mean that the values of the three elements of the triplets are the same. The expression $A =_{NL2} B$ will mean the same thing, except in NL2, as will $=_{INL1}$ in INL1, $=_{INL2}$ in INL2 and $=_{PNL1}$ in PNL1.

Theorem 4.2.4:
a) Let A be an expression in NL1. Then $\neg\neg A =_{NL1} A$.
b) Let A be an expression in NL2. Then $\neg\neg A =_{NL2} A$.
c) Let A be an expression in INL1. Then $\neg\neg A =_{INL1} A$.
d) Let A be an expression in INL2. Then $\neg\neg A =_{INL2} A$.
e) Let A be an expression in PNL1. Then $\neg\neg A =_{PNL1} A$.

Proof:
By the definitions, in all cases the $\neg$ operator interchanges the values of t and f. Therefore, interchanging them twice will return the original.

Theorem 4.2.5: If A and B are expressions in NL1, then

i) $A \wedge B =_{NL1} B \wedge A$.
ii) $A \vee B =_{NL1} B \vee A$.

In other words, $\wedge$ and $\vee$ are commutative.

Proof: Let $A = (t_1, i_1, f_1)$ and $B = (t_2, i_2, f_2)$.
i) $(t_1, i_1, f_1) \wedge (t_2, i_2, f_2) = (\min \{ t_1, t_2\}, 1 - (\min \{ t_1, t_2\} + \max \{ f_1, f_2 \}),$
$\max \{ f_1, f_2 \}) =$

$(\min \{ t_2, t_1\}, 1 - (\min \{ t_2, t_1\} + \max \{ f_2, f_1 \}), ,$
$\max \{ f_2, f_1 \}) = (t_2, i_2, f_2) \wedge (t_1, i_1, f_1).$

Therefore, $\wedge$ is commutative.

ii) $(t_1, i_1, f_1) \vee (t_2, i_2, f_2) = (\max \{ t_1, t_2\}, 1 - (\max \{ t_1, t_2\} + \min \{ f_1, f_2 \}), , \min \{ f_1, f_2 \}) =$

$(\max \{ t_2, t_1\}, 1 - (\max \{ t_2, t_1\} + \min \{ f_2, f_1 \}), , \min \{ f_2, f_1 \}) = (t_2, i_2, f_2) \vee (t_1, i_1, f_1).$

Therefore, $\vee$ is commutative.

Theorem 4.2.6: If A and B are expressions in NL2, then

i) $A \wedge B =_{NL2} B \wedge A$.
ii) $A \vee B =_{NL2} B \vee A$.

In other words, $\wedge$ and $\vee$ are commutative in NL2.

Proof: Similar to that of theorem 4.2.5.

Theorem 4.2.7: If A and B are expressions in INL1, INL2 or PNL1, then

i) $A \wedge B =_{INL1} B \wedge A$.
ii) $A \vee B =_{INL1} B \vee A$.
iii) $A \wedge B =_{INL2} B \wedge A$.
iv) $A \vee B =_{INL2} B \vee A$.
v) $A \wedge B =_{PNL1} B \wedge A$.
vi) $A \vee B =_{PNL1} B \vee A$.

In other words, $\wedge$ and $\vee$ are commutative in INL1, INL2 and PNL1.

Proof: Since $\min\{x, y\} = \min\{y, x\}$ for all numbers x and y, the results are a direct result of the definitions of the operators.

Theorem 4.2.8: If A, B and C are expressions in NL1, then :
i) $(A \wedge B) \wedge C =_{NL1} A \wedge (B \wedge C)$.
ii) $(A \vee B) \vee C =_{NL1} A \vee (B \vee C)$.

Proof: Let $A = (t_1, i_1, f_1)$, $B = (t_2, i_2, f_2)$ and $C = (t_3, i_3, f_3)$.
i) $[(t_1, i_1, f_1) \wedge (t_2, i_2, f_2)] \wedge (t_3, i_3, f_3) =$

$[(\min\{t_1, t_2\}, 1 - (\min\{t_1, t_2\} + \max\{t_1, t_2\}), \max\{t_1, t_2\}] \wedge (t_3, i_3, f_3) =$

$(\min\{\min\{t_1, t_2\}, t_3\}, 1 - (\min\{\min\{t_1, t_2\}, t_3\} + \max\{\max\{t_1, t_2\}, t_3\}),$

$\max\{\max\{t_1, t_2\}, t_3\}) =$

$(\min\{t_1, t_2, t_3\}, 1 - (\min\{t_1, t_2, t_3\} + \max\{t_1, t_2, t_3\}), \max\{t_1, t_2, t_3\}) =$

$(\min\{t_1, \min\{t_2, t_3\}\}, 1 - (\min\{t_1, \min\{t_2, t_3\}\} + \max\{t_1, \max\{t_2, t_3\}\}),$

$\max\{t_1, \max\{t_2, t_3\}\}) = (t_1, i_1, f_1) \wedge [(t_2, i_2, f_2) \wedge (t_3, i_3, f_3)]$

Therefore, the $\wedge$ connective is associative.

ii) The proof that $\vee$ is associative is similar and will not be done.

Corollary: If $\{A_1, A_2, A_3, \ldots, A_n\}$ is a set of elements in NL1, then

$A_1 \vee A_2 \vee A_3 \vee \ldots \vee A_n = (t, i, f)$

where $t = \max\{ t_1, t_2, t_3, \ldots, t_n \}$ and $f = \min\{ f_1, f_2, f_3, \ldots, f_n \}$.

Proof: Using the results of theorem 4.2.8, we can associate the elements any way we wish without changing the value. Therefore, we will left associate everything. In other words, we will write the expression in the form

$(\ldots (A_1 \vee A_2) \vee A_3) \vee \ldots) \vee A_n)$

The proof will be by induction on $k \geq 3$.

Basis: $k = 3$.

$((A_1 \vee A_2) \vee A_3) = (t, i, f)$

where $t = \max\{ \max\{ t_1, t_2 \}, t_3 \} = \max\{ t_1, t_2, t_3 \}$

and

$f = \max\{ \max\{ f_1, f_2 \}, f_3 \} = \max\{ f_1, f_2, f_3 \}$.

Inductive step: Assume that for $k > 3$

$(\ldots (A_1 \vee A_2) \vee A_3) \vee \ldots) \vee A_k) = (t, i, f)$ where

$t = \max\{ t_1, t_2, t_3, \ldots, t_k \}$ and $f = \min\{ f_1, f_2, f_3, \ldots, f_k \}$.

Then, by definition

$(\ldots (A_1 \vee A_2) \vee A_3) \vee \ldots) \vee A_k) \vee A_{k+1}) = (t_s, i_s, f_s)$ where

$t_s = \max\{ t, t_{k+1} \} = \max\{ \max\{t_1, t_2, \ldots, t_k\}, t_{k+1} \} = \max\{ t_1, t_2, \ldots, t_k, t_{k+1} \}$

$f_s = \min\{ f, f_{k+1} \} = \min\{ \min\{f_1, f_2, \ldots, f_k\}, f_{k+1} \} = \min\{ f_1, f_2, \ldots, f_k, f_{k+1} \}$

Therefore, by the principle of mathematical induction, the formula is true for all n.

Corollary: If $\{ A_1, A_2, A_3, \ldots, A_n \}$ is a set of elements in NL1, then

$A_1 \wedge A_2 \wedge A_3 \wedge \ldots \wedge A_n = (t, i, f)$

where $t = \min\{ t_1, t_2, t_3, \ldots, t_n \}$ and $f = \max\{ f_1, f_2, f_3, \ldots, f_n \}$.

Proof: Similar to that of the previous corollary and will not be done.

Theorem 4.2.9: If A, B and C are expressions in NL2, then :

i) $(A \wedge B) \wedge C =_{NL2} A \wedge (B \wedge C)$.

ii) $(A \vee B) \vee C =_{NL2} A \vee (B \vee C)$.

Proof: Let $A = (t_1, i_1, f_1)$, $B = (t_2, i_2, f_2)$ and $C = (t_3, i_3, f_3)$.
i) Clearly, $\max \{ \max \{ i_1, i_2 \}, i_3 \} = \max \{ i_1, \max\{ i_2, i_3 \} \}$, so the middle term will be the same for both sides. Therefore, if the values of the third terms can be proven to be equal, the proof is complete.

On the left side, the value of the third entry of $A \wedge B$ is

$\min\{ 1 - \max\{i_1, i_2 \}, \max \{ f_1, f_2 \} \}$

therefore, the third entry of $(A \wedge B) \wedge C$ is

$\min \{ 1 - \max \{ i_1, i_2, i_3 \}, \max \{ \min\{ 1 - \max\{i_1, i_2 \}, \max \{ f_1, f_2 \} \}, f_3 \} \}$.

On the right side, the third entry of $B \wedge C$ is

$\min\{ 1 - \max\{i_2, i_3 \}, \max \{ f_2, f_3 \} \}$

therefore, the third entry of $A \wedge (B \wedge C)$ is

$\min \{ 1 - \max \{ i_1, i_2, i_3 \}, \max \{ \min\{ 1 - \max\{i_2, i_3 \}, \max \{ f_2, f_3 \} \}, f_1 \} \}$.

If the smallest value of each min expression is $1 - \max \{ i_1, i_2, i_3 \}$, then the values are the same. Therefore, assume that the value of one of the third entries is not $1 - \max \{ i_1, i_2, i_3 \}$, which means that it is either f_1, f_2, or f_3, as the minimum of $1 - \max \{ i_1, i_2, i_3 \}$ and $1 - \max\{i_2, i_3 \}$ is $1 - \max \{ i_1, i_2, i_3 \}$.

Case 1: Assume that

$f_1 = \min \{ 1 - \max \{ i_1, i_2, i_3 \}, \max \{ \min\{ 1 - \max\{i_1, i_2 \}, \max \{ f_1, f_2 \} \}, f_3 \}$.

This means that $f_1 \geq f_2$, $f_1 \leq 1 - \max\{i_1, i_2 \}$, $f_1 \geq f_3$ and $f_1 \leq 1 - \max \{ i_1, i_2, i_3 \}$. With these restrictions, f_1 is greater than or equal to whatever value emerges from

$\min\{ 1 - \max\{i_2, i_3 \}, \max \{ f_2, f_3 \} \}$

so

$f_1 = \max \{ \min\{ 1 - \max\{i_2, i_3 \}, \max \{ f_2, f_3 \} \}, f_1 \}$.

Since $f_1 < 1 - \max \{ i_1, i_2, i_3 \}$, the third term of the right side must also be f_1.

Case 2 Assume that

$f_2 = \min \{ 1 - \max \{ i_1, i_2, i_3 \}, \max \{ \min\{ 1 - \max\{i_1, i_2 \}, \max \{ f_1, f_2 \} \}, f_3 \}$.

This means that $f_2 \geq f_1$, $f_2 \leq 1 - \max\{i_1, i_2\}$, $f_2 \geq f_3$ and $f_2 \leq 1 - \max\{ i_1, i_2, i_3 \}$. With these restrictions

$$f_2 = \min\{ 1 - \max\{i_2, i_3\}, \max\{ f_2, f_3 \} \}$$

$$f_2 = \max\{ f_2, f_1 \}$$

and $f_2 = \min\{ 1 - \max\{ i_1, i_2, i_3 \}, f_2 \}$. Therefore, the value of the third term on the right side is also f_2.

The proofs of the remaining cases are all similar and will be omitted. Therefore, the conjunction is associative in NL2.

ii) The proofs for the associativity of the disjunction are similar, only the testing is done on the first term of the triplet rather than the third.

Corollary: If $\{ A_1, A_2, A_3, \ldots, A_n \}$ is a set of elements in NL2, then

$A_1 \wedge A_2 \wedge A_3 \wedge \ldots \wedge A_n = (t, i, f)$

where $i = \max\{ i_1, i_2, i_3, \ldots, i_n \}$ and

$f = \min\{ 1 - \max\{ i_1, i_2, i_3, \ldots, i_n \}, \max\{ f_1, f_2, f_3, \ldots, f_n \}\}$.

Corollary: If $\{ A_1, A_2, A_3, \ldots, A_n \}$ is a set of elements in NL2, then

$A_1 \vee A_2 \vee A_3 \vee \ldots \vee A_n = (t, i, f)$

where $i = \max\{ i_1, i_2, i_3, \ldots, i_n \}$ and

$t = \min\{ 1 - i, \max\{t_1, t_2, t_3, \ldots, t_n \}\}$.

The previous four corollaries allow for the definition of the universal and existential quantifiers in NL1 and NL2.

Definition 4.2.7: Let

$$\bigwedge_{i \in U} (t_I, i_I, f_I)$$

be a conjunction of all of the values in the universal set of an implementation of NL1. If

$t_{min} = \min\{ t_1, \ldots, t_k, \ldots \}$
$f_{max} = \max\{ f_1, \ldots, f_k, \ldots \}$

then the value of the universal quantifier in NL1 is

$$(t_{min}, 1 - (t_{min}+f_{max}), f_{max}).$$

If the values of t_{min} or f_{max} are within certain ranges, we can then define the universal quantifier for NL1 as one where all values of the expression are greater than or less than a specific value. In most cases, qualifications will be placed on the results.

Example:
If NL_K is an implementation of Neutrosophic logic where the connectives are defined as they are in NL1 and the indeterminate value of the elements of NL_K is never greater than 0.10, then the universal statement

$$(x)\ x \varepsilon NL_K\ (x \vee \neg x)$$

would have a truth value of at least 0.45.

Definition 4.2.8: Let

$$\bigvee_{i \varepsilon U} (t_i, i_i, f_i)$$

be a disjunction of all of the values in the universal set of an implementation of NL1. If

$t_{max} = \max \{ t_1, \ldots, t_k, \ldots \}$
$f_{min} = \min \{ f_1, \ldots, f_k, \ldots \}$

then the value of the existential quantifier in NL1 is given by

$$(t_{max}, 1 - (t_{max}+f_{min}), f_{min}).$$

Definition 4.2.9: Let

$$\bigwedge_{i \varepsilon U} (t_I, i_I, f_I)$$

be a conjunction of all of the values in the universal set of an implementation of NL2. If

$i_{max} = \max \{ i_1, \ldots, i_k, \ldots \}$
$f_{max} = \max \{ f_1, \ldots, f_k, \ldots \}$

then the value of the universal quantifier in NL2 is

$$(1-(i + f), i = i_{max}, f = \min \{ 1 - i_{max}, f_{max} \}).$$

Definition 4.2.10: Let

$$\bigvee_{i \,\varepsilon\, U} (t_i, i_i, f_i)$$

be a disjunction of all of the values in the universal set of an implementation of NL2. If

$t_{max} = \max \{ t_1, \ldots, t_k, \ldots \}$
$i_{max} = \max \{ i_1, \ldots, i_k, \ldots \}$

then the value of the existential quantifier in NL2 is given by

$(t = \min \{ 1 - i_{max}, t_{max} \}, i = i_{max}, f = 1 - (t + i))$.

Theorem 4.2.10: If A, B and C are elements in INL1, INL2 or PNL1, then

i) $(A \wedge B) \wedge C =_{INL1} A \wedge (B \wedge C)$.
ii) $(A \vee B) \vee C =_{INL1} A \vee (B \vee C)$.
iii) $(A \wedge B) \wedge C =_{INL2} A \wedge (B \wedge C)$.
iv) $(A \vee B) \vee C =_{INL2} A \vee (B \vee C)$.
v) $(A \wedge B) \wedge C =_{PNL1} A \wedge (B \wedge C)$.
vi) $(A \vee B) \vee C =_{PNL1} A \vee (B \vee C)$.

Proof: Since $\min\{\min \{x, y\}, z\} = \min\{x, \min\{y, z\}\}$ and $\max\{\max \{x, y\}, z\} = \max\{x, \max\{y, z\}\}$ for all numbers x, y and z, all results follow directly from the definitions of the operators.

Corollary: If $\{ A_1, A_2, A_3, \ldots, A_n \}$ is a set of elements in INL1, then

$A_1 \wedge A_2 \wedge A_3 \wedge \ldots \wedge A_n = (t, i, f, u)$

where $t = \min\{ t_1, \ldots, t_n \}$, $i = \min\{ i_1, \ldots, i_n \}$, $f = \max\{ f_1, \ldots, f_n \}$ and $u = 1 - t - i - f$.

Corollary: If $\{ A_1, A_2, A_3, \ldots, A_n \}$ is a set of elements in INL1, then

$A_1 \vee A_2 \vee A_3 \vee \ldots \vee A_n = (t, i, f, u)$

where $t = \max\{ t_1, \ldots, t_n \}$, $i = \min\{ i_1, \ldots, i_n \}$, $f = \min\{ f_1, \ldots, f_n \}$ and $u = 1 - t - i - f$.

Corollary: If $\{ A_1, A_2, A_3, \ldots, A_n \}$ is a set of elements in INL2, then

$A_1 \wedge A_2 \wedge A_3 \wedge \ldots \wedge A_n = (t, i, f, u)$

where $t = \min\{ t_1, \ldots, t_n \}$, $f = \max\{ f_1, \ldots, f_n \}$, $u = \min\{ u_1, \ldots, u_n \}$ and

$i = 1 - t - f - u$.

Corollary: If $\{ A_1, A_2, A_3, \ldots, A_n \}$ is a set of elements in INL2, then

$A_1 \vee A_2 \vee A_3 \vee \ldots \vee A_n = (t, i, f, u)$

where $t = \max\{ t_1, \ldots, t_n \}$, $f = \min\{ f_1, \ldots, f_n \}$, $u = \min\{ u_1, \ldots, u_n \}$ and $i = 1 - t - f - u$.

Corollary: If $\{ A_1, A_2, A_3, \ldots, A_n \}$ is a set of elements in PNL1, then

$A_1 \wedge A_2 \wedge A_3 \wedge \ldots \wedge A_n = (t, i, f)$

where $t = \min\{ t_1, \ldots, t_n \}$, $i = \max\{ i_1, \ldots, i_n \}$ and $f = \max\{ f_1, \ldots, f_n \}$.

Corollary: If $\{ A_1, A_2, A_3, \ldots, A_n \}$ is a set of elements in PNL1, then

$A_1 \vee A_2 \ A_3 \vee \ldots \vee A_n = (t, i, f)$

where $t = \max\{ t_1, \ldots, t_n \}$, $i = \max\{ i_1, \ldots, i_n \}$ and $f = \min\{ f_1, \ldots, f_n \}$.

Each of the previous six corollaries can be proven by an induction on the number of elements in the conjunction or disjunction.

Armed with these corollaries, it is then possible to define the universal and existential quantifiers for INL1, INL2 and PNL1.

Definition 4.2.11: Let

$$\bigwedge_{i \, \varepsilon \, U} (t_i, i_i, f_i, u_i)$$

be a conjunction of all of the values in the universal set of an implementation of INL1. If

$t_{min} = \min \{ t_1, \ldots, t_k, \ldots \}$
$i_{min} = \min \{ i_1, \ldots., i_k, \ldots \}$
$f_{max} = \max \{ f_1, \ldots, f_k, \ldots \}$

then the value of the universal quantifier in INL1 is

$(t_{min}, i_{min}, f_{max}, u = 1 - t_{min} - i_{min} - f_{max})$.

Definition 4.2.12: Let

$$\bigwedge_{i \in U} (t_i, i_i, f_i, u_i)$$

be a conjunction of all of the values in the universal set of an implementation of INL2. If

$t_{min} = \min \{ t_1, \ldots, t_k, \ldots \}$
$f_{max} = \max \{ f_1, \ldots, f_k, \ldots \}$
$u_{min} = \min \{ u_1, \ldots, u_k, \ldots \}$

then the value of the universal quantifier in NL1 is

$$(t_{min}, 1 - t_{min} - f_{max} - u_{min}, f_{max}, u_{min}).$$

Definition 4.2.13: Let

$$\bigwedge_{i \in U} (t_i, i_i, f_i)$$

be a conjunction of all of the values in the universal set of an implementation of PNL1. If

$t_{min} = \min \{ t_1, \ldots, t_k, \ldots \}$
$i_{max} = \max \{ i_1, \ldots, I_k, \ldots \}$
$f_{max} = \max \{ f_1, \ldots, f_k, \ldots \}$

then the value of the universal quantifier in PNL1 is

$$(t_{min}, i_{max}, f_{max}).$$

Definition 4.2.14: Let

$$\bigvee_{i \in U} (t_i, i_i, f_i, u_i)$$

be a disjunction of all of the values in the universal set of an implementation of INL1. If

$t_{max} = \max \{ t_1, \ldots, t_k, \ldots \}$
$i_{min} = \min \{ i_1, \ldots., i_k, \ldots \}$
$f_{min} = \min \{ f_1, \ldots, f_k, \ldots \}$

then the value of the existential quantifier in INL1 is

$$(t_{max}, i_{min}, f_{min}, u = 1 - t_{max} - i_{min} - f_{min}).$$

Definition 4.2.15: Let

$$\bigvee_{i \in U} (t_i, i_i, f_i, u_i)$$

be a disjunction of all of the values in the universal set of an implementation of INL2. If

$t_{max} = \max \{ t_1, \ldots, t_k, \ldots \}$
$f_{min} = \min \{ f_1, \ldots, f_k, \ldots \}$
$u_{min} = \min \{ u_1, \ldots, u_k, \ldots \}$

then the value of the universal quantifier in NL1 is

$$(t_{max}, 1 - t_{max} - f_{min} - u_{min}, f_{min}, u_{min}).$$

Definition 4.2.16: Let

$$\bigvee_{i \in U} (t_i, i_i, f_i)$$

be a disjunction of all of the values in the universal set of an implementation of PNL1. If

$t_{max} = \max \{ t_1, \ldots, t_k, \ldots \}$
$i_{max} = \max \{ i_1, \ldots, I_k, \ldots \}$
$f_{min} = \min \{ f_1, \ldots, f_k, \ldots \}$

then the value of the universal quantifier in PNL1 is

$$(t_{max}, i_{max}, f_{min}).$$

Section 3
Algebraic Properties of Neutrosophic Logics

Definition 4.3.1: A semigroup is a set S and a binary operation ○ that is closed with respect to S and associative on S.

Theorem 4.3.1:
(i) The set of triplets (t, i, f) where t + i + f = 1.0 with the NL1 definition of $\wedge$ is a semigroup.

(ii) The set of triplets (t, i, f) where t + i + f = 1.0 with the NL1 definition of $\vee$ is a semigroup.

(iii) The set of triplets (t, i, f) where t + i + f = 1.0 with the NL2 definition of $\wedge$ is a semigroup.

(iv) The set of triplets (t, i, f) where t + i + f = 1.0 with the NL2 definition of $\vee$ is a semigroup.

(v) The set of 4-tuples (t, i, f, u) where t + i + f + u = 1.0 with the INL1 definition of $\wedge$ is a semigroup.

(vi) The set of 4-tuples (t, i, f, u) where t + i + f + u = 1.0 with the INL1 definition of $\vee$ is a semigroup.

(vii) The set of 4-tuples (t, i, f, u) where t + i + f + u = 1.0 with the INL2 definition of $\wedge$ is a semigroup.

(viii) The set of 4-tuples (t, i, f, u) where t + i + f + u = 1.0 with the INL2 definition of $\vee$ is a semigroup.

(ix) The set of 3-tuples (t, i, f) where t + i + f $\geq$ 1.0 with the PNL1 definition of $\wedge$ is a semigroup.

(x) The set of 3-tuples (t, i, f) where t + i + f $\geq$ 1.0 with the PNL1 definition of $\wedge$ is a semigroup.

Proof: Since a semigroup is a set closed on the operation where the associative property holds for all elements, all ten results are direct consequences of the previous theorems where the associative property was verified for all the logics.

Theorem 4.3.2: If A and B are expressions in NL1, then

i) $(A \vee B) \wedge A =_{NL1} A$.
ii) $(A \wedge B) \vee A =_{NL1} A$.

In other words, $\vee$ and $\wedge$ satisfy the absorption laws.

Proof: Let $A = (t_1, i_1, f_1)$ and $B = (t_2, i_2, f_2)$. The proof is by examining the elements of the triplets one at a time.

(i) **Case 1:** Examining the first element of the triplet.

The values of the first elements of the triplets on the left side is computed by the following formula

$$\min\{ \max\{ t_1, t_2 \}, t_1 \}$$

Subcase 1: $t_1 > t_2$

Then,

$\min\{ \max\{ t_1, t_2 \}, t_1 \} = \min\{ t_1, t_1 \} = t_1$

Subcase 2: $t_1 < t_2$

Then,

$\min\{ \max\{ t_1, t_2 \}, t_1 \} = \min\{ t_2, t_1 \} = t_1$

Therefore, the first elements of the triplets are the same in all cases.

(ii) **Case 2:** Examining the third element of the triplet.

The values of the third elements of the triplets on the left side is computed by the following formula

$\max\{ \min\{ f_1, f_2 \}, f_1 \}$

Subcase 1: $f_1 > f_2$

Then,

$\max\{ \min\{ f_1, f_2 \}, f_1 \} = \max\{ f_2, f_1 \} = f_1$

Subcase 2: $f_1 < f_2$

Then,

$\max\{ \min\{ f_1, f_2 \}, f_1 \} = \max\{ f_1, f_1 \} = f_1$

Therefore, the third elements of the triplets are the same in all cases.

Since the first and third elements of the triplets are the same in all cases, the formula is true.

(ii) **Case 1:** The first elements of the triplets

The values of the first elements of the triplets on the left side is computed by the following formula

$\max\{ \min\{ t_1, t_2 \}, t_1 \}$.

Subcase 1: $t_1 > t_2$

Then,

$$\max\{ \min\{ t_1, t_2 \}, t_1 \} = \max\{ t_2, t_1 \} = t_1 .$$

Subcase 2: $t_1 < t_2$

Then,

$$\max\{ \min\{ t_1, t_2 \}, t_1 \} = \max\{ t_1, t_1 \} = t_1.$$

Therefore, the first elements are the triplets are the same in all cases.

Case 2: The third elements of the triplets

The values of the third elements of the triplets on the left side is computed by the following formula

$$\max\{ \min\{ f_1, f_2 \}, f_1 \}$$

Subcase 1: $f_1 > f_2$

Then,

$$\max\{ \min\{ f_1, f_2 \}, f_1 \} = \max\{ f_2, f_1 \} = f_1$$

Subcase 2: : $f_1 < f_2$

Then,

$$\max\{ \min\{ f_1, f_2 \}, f_1 \} = \max\{ f_1, f_1 \} = f_1$$

Therefore, the third elements of the triplets are the same in all cases.

Since the first and third elements of the triplets are the same in all cases, the formula is true.

Theorem 4.3.3: If A and B are expressions in NL2, then

i) $(A \vee B) \wedge A \neq_{NL2} A.$
ii) $(A \wedge B) \vee A \neq_{NL2} A.$

In other words, $\vee$ and $\wedge$ do not satisfy the absorption laws.

Proof: Let $A = (t_1, i_1, f_1)$ and $B = (t_2, i_2, f_2)$.

i) If $i_2 > i_1$ then the middle term of $A \vee B$ is i_2 and the middle term of $(A \vee B) \wedge A$ is also i_2. Therefore, $(A \vee B) \wedge A \neq_{NL2} A.$

ii) If $i_2 > i_1$ then the middle term of $A \wedge B$ is i_2 and the middle term of $(A \wedge B) \vee A$ is also i_2. Therefore, $(A \vee B) \wedge A \neq_{NL2} A$.

Theorem 4.3.4: INL1 and INL2 do not satisfy either absorption law.

Proof: Let A = (0.10, 0.20, 0.30, 0.40) and B = (0.10, 0.10, 0.30, 0.50) and elements of INL1. Then $A \vee B$ = (0.10, 0.10, 0.30, 0.50) and

(0.10, 0.10, 0.30, 0.50) $\wedge$ (0.10, 0.20, 0.30, 0.40) = (0.10, 0.10, 0.30, 0.50), which is not A.

$A \wedge B$ = 0.10, 0.10, 0.30, 0.50) AND

0.10, 0.10, 0.30, 0.50) $\vee$ (0.10, 0.20, 0.30, 0.40) = (0.10, 0.10, 0.30, 0.50) which is also not A.

Let A = (0.10, 0.40, 0.30, 0.20) and B = (0.10, 0.50, 0.30, 0.10) be elements of INL2. Then $A \vee B$ = (0.10, 0.50 , 0.30, 0.10) and

(0.10, 0.50, 0.30, 0.10) $\wedge$ (0.10, 0.40, 0.30, 0.20) = (0.10, 0.50 , 0.30, 0.10) which is not A.

$A \wedge B$ = (0.10, 0.50 , 0.30, 0.10) and

(0.10, 0.50 , 0.30, 0.10) $\vee$ (0.10, 0.40, 0.30, 0.20) = (0.10, 0.50, 0.30, 0.10) which is not A.

Theorem 4.3.5: PNL1 does not satisfy either absorption law.

Proof: Let A = (0.10, 0.10, 0.90) and B = (0.10, 0.20, 0.90) be elements in PNL1. Then $A \vee B$ = (0.10, 0.20, 0.90) and

(0.10, 0.20, 0.90) $\wedge$ (0.10, 0.10, 0.90) = (0.10, 0.20, 0.90) which is not A.

$A \wedge B$ = (0.10, 0.20, 0.90) and

(0.10, 0.20, 0.90) $\vee$ (0.10, 0.10, 0.90) = (0.10, 0.20, 0.90) which is not A.

Theorem 4.3.6: If A is an element of NL1, then $(1, 0, 0) \wedge A =_{NL1} A \wedge (1, 0, 0) =_{NL1} A$ and there is no other element that is an identity for $\wedge$.

Proof: Let $A = (t_1, i_1, f_1)$. Then $(1, 0, 0) \wedge (t_1, i_1, f_1) =$

$(\min \{1, t_1 \}, \ 1 - (\min \{1, t_1 \} + \max \{ 0, f_1 \}), \max \{ 0, f_1 \}) = (t_1, i_1, f_1)$.

It has already been proven that $\wedge$ is commutative, so (1, 0, 0) is a commutative identity for $\wedge$.

Suppose that there is another element (t, i, f) that is an identity for $\wedge$. By the limitations on the values, t < 1.00. It is then possible to choose a value t_k such that $t < t_k < 1$. Performing the operation (t, i, f) $\wedge$ (t_k, i_k, f_k), the result in the first position is t, since it is the smaller. The triple then cannot be equal to (t_k, i_k, f_k), violating the assumption that (t, i, f) is an identity element for $\wedge$.

Definition 4.3.2: A monoid is a semigroup S with an identity element.

Theorem 4.3.7: The algebra [NL1, $\wedge$] is a monoid.

Proof: Since a monoid is a semigroup with an identity, this is a direct consequence of the previous theorem and the previous result that [NL1, $\wedge$] is a semigroup.

Theorem 4.3.8: Let A be an element of NL1. Then

(0, 0, 1) $\vee$ A $=_{NL1}$ A $\vee$ (0, 0, 1) $=_{NL1}$ A

and there is no other such element that is an identity for $\vee$.

Proof: Let A = (t_1, i_1, f_1).

Then (0, 0, 1) $\vee$ (t_1, i_1, f_1) = (max { 0, t_1 }, 1 – (max { 0, t_1 } + min { 1, f_1 }) , min { 1, f_1 }) = (t_1, i_1, f_1)

and the rest follows from the commutativity of $\vee$.

Assume that there is another element (t, i, f) that is an identity for $\vee$. Again, by the limitations on the values, t < 1.00. It is then possible to choose a value f_k such that $f < f_k < 1.00$. Performing the operation, (t, i, f) $\vee$ (t_k, i_k, f_k) the result in the last position is f, since it is the smallest. The triple then cannot be equal to (t_k, i_k, f_k), violating the assumption that (t, i, f) is an identity element for $\vee$.

Theorem 4.3.9: The algebra [NL1, $\vee$] is a monoid.

Proof: A direct consequence of the definition of a monoid, the previous theorem and the earlier result that [NL1, $\vee$] is a semigroup.

Theorem 4.3.10: If A is an arbitrary element of NL2, then there is no element (t, i, f) such that (t, i, f) $\wedge$ A $=_{NL2}$ A $\wedge$ (t, i, f) $=_{NL2}$ A. Therefore, there is no identity element for $\wedge$ in NL2 and [NL2, $\wedge$] is not a monoid.

Proof: Let A = (t_1, i_1, f_1). Then (t, i, f) $\wedge$ (t_1, i_1, f_1) =

(1 – max { i, i_1 } – min { 1 - max { i, i_1 }, max { f, f_1 }}, max { i, i_1 },
min { 1 - max { i, i_1 }, max { f, f_1 }}).

The only way that the middle term can be i_1 is if i = 0 and the only way the third term can be f_1 is if f = 0. This forces t to be 1.0. However, min { 1 – i_1, f_1 } is not f_1 in general.

Theorem 4.3.11: If A is an arbitrary element of NL2, then there is no element (t, i, f) such that (t, i, f) $\wedge$ A $=_{NL2}$ A $\vee$ (t, i, f) $=_{NL2}$ A. Therefore, there is no identity element for $\vee$ in NL2 and [NL2, $\vee$] is not a monoid.

Proof: Similar to that of theorem 4.3.10.

Theorem 4.3.12:
a) There is no element in INL1 that is an identity for $\wedge$.
b) There is no element in INL1 that is an identity for $\vee$.

Therefore, neither [INL1, $\wedge$] or [INL1, $\vee$] is a monoid.

Proof:
a) Let A = (t_1, i_1, f_1, u_1) be an element of INL1. For there to be an element I = (t, i, f, u) such that A $\wedge$ I = A, it follows by definition that t $\geq$ t_1 and i $\geq$ i_1 for all values of t_1 and i_1. The only values that satisfy this are t = 1.0 and i = 1.0. However, the constraints on the values disallow the possibility that they are simultaneously one.

b) The proof is similar, except f is used rather than t.

Theorem 4.3.13:
a) There is no element in INL2 that is an identity for $\wedge$.
b) There is no element in INL2 that is an identity for $\vee$.

Therefore, neither [INL2, $\wedge$] or [INL2, $\vee$] is a monoid.

Proof: For $\wedge$, apply the reasoning of theorem 4.3.12 using t and u and for $\vee$ apply the reasoning using f and u.

Theorem 4.3.14:
a) If there is a maximum value (maxt) that the true component can have in PNL1, then (maxt, 0, 0) is an identity for $\wedge$.
b) If there is a maximum value (maxf) that the false component can have in PNL1, then (0,0,maxf) is an identity for $\vee$.

Therefore, if there is a maximum value for the true component, [PNL1, $\wedge$] is a monoid and if there is a maximum value for the false component, [PNL1, $\vee$] is a monoid.

Proof: Let A = (t_1, i_1, f_1) be an element of PNL1. Then

$(t_1, i_1, f_1) \wedge (maxt, 0, 0) = (\min\{t_1, maxt\}, \max\{i_1, 0\}, \max\{f_1, 0\}) = (t_1, i_1, f_1)$.

AND

$(t_1, i_1, f_1) \vee (0, 0, maxf) = (\max\{t_1, 0\}, \max\{i_1, 0\}, \min\{f_1, maxf\}) = (t_1, i_1, f_1)$.

Clearly, if the value of the true component is unbounded, it is not possible to find a value for true where the minimum of that value and all others is always the other value. A similar situation holds for the values of the false component.

Definition 4.3.3: Let o be a closed binary operation on a set S. The operation is said to satisfy the **idempotent** law if x o x = x for all x ϵ S.

Theorem 4.3.15: For all (t, i, f) ε NL1, (t, i, f) $\wedge$ (t, i, f) = (t, i, f) and (t, i, f) $\vee$ (t, i, f) = (t, i, f). In other words, $\vee$ and $\wedge$ are idempotent in NL1.

Proof: Since min { x, x } = x and max{ x, x } = x, the conclusions follow from the definitions of the operators.

Theorem 4.3.16: For all (t, i, f) ε NL2, (t, i, f) $\wedge$ (t, i, f) = (t, i, f) and (t, i, f) $\vee$ (t, i, f) = (t, i, f). In other words, $\wedge$ and $\vee$ are idempotent in NL2.

Proof: By definition, (t, i, f) $\wedge$ (t, i, f) =
(1 – max { i, i } – min { 1 – max { i,i }, max { f, f} }, max { i, i},
min { 1 – max {i,i }, max {f, f}}).

The middle term is clearly i and with the restriction that t + f + i = 1.0, it follows that max {f, f } = f, max { i, i } = i and 1 – i can never be less than f. Therefore the third term is f, so the first term must be t.

By definition, (t, i, f) $\wedge$ (t, i, f) = (min { 1 – max { i, i }, max { t,t }}, max {i, i },
1 – max {i, i } – min { 1 – max { i, i }, max { t, t }}).

The middle term is clearly i and with the restriction that t + f + i = 1.0, it follows that max {t, t } = t, max { i, i } = i and 1 – i can never be less than t. Therefore the first term is t, so the third term must be f.

Theorem 4.3.17:
a) $\wedge$ and $\vee$ are idempotent in INL1.
b) $\wedge$ and $\vee$ are idempotent in INL2.
c) $\wedge$ and $\vee$ are idempotent in PNL1.

Proof:
All of the proofs are based on the principles that max {x, x } = x and min{x, x} = x for all numbers x.

Theorem 4.3.18: Let A, B and C be elements in NL1. The Distributive law holds for $\wedge$ over $\vee$, in other words

$A \wedge (B \vee C) =_{NL1} (A \wedge B) \vee (A \wedge C)$.

Proof: Let A= (t_1, i_1, f_1), B = (t_2, i_2, f_2) and C = (t_3, i_3, f_3). In this case, the proof will be by examining the values of the components of the triplets generated by the operations.

Case 1: The values of the first element in the ordered triplet.
On the left side, we would have

$\min\{ t_1, \max\{ t_2, t_3 \} \}$

and on the right side we would have

$\max \{ \min\{ t_1, t_2 \}, \min\{ t_1, t_3 \} \}$.

Subcase 1: t_1 is less than or equal to t_2 and t_3.

Then

$\min\{ t_1, \max\{ t_2, t_3 \} \} = t_1 = \max\{ t_1, t_1 \} = \max \{ \min\{ t_1, t_2 \}, \min\{ t_1, t_3 \} \}$.

Subcase 2: $t_2 \geq t_1 \geq t_3$

Then

$\min\{ t_1, \max\{ t_2, t_3 \} \} = \min\{ t_1, t_2 \} = t_1 = \max\{ t_1, t_3 \} = \max \{ \min\{ t_1, t_2 \}, \min\{ t_1, t_3 \} \}$.

Subcase 3: $t_3 \geq t_1 \geq t_2$

Then

$\min\{ t_1, \max\{ t_2, t_3 \} \} = \min\{ t_1, t_3 \} = t_1 = \max\{ t_2, t_1 \} = \max \{ \min\{ t_1, t_2 \}, \min\{ t_1, t_3 \} \}$.

Subcase 4: $t_1 \geq t_2 \geq t_3$

Then

$\min\{ t_1, \max\{ t_2, t_3 \} \} = t_2 = \max\{ t_2 , t_3 \} = \max \{ \min\{ t_1, t_2 \}, \min\{ t_1, t_3 \} \}$.

Subcase 5: $t_1 \geq t_3 \geq t_2$

Then

$\min\{ t_1, \max\{ t_2, t_3 \} \} = t_3 = \max\{ t_2, t_3 \} = \max \{ \min\{ t_1, t_2 \}, \min\{ t_1, t_3 \} \}$.

All cases have been considered, so the first element of the triplet is the same on both sides of the expression.

Case 2: The values of the third element in the ordered triplet.

In this case, the third element on the left side is computed by

$\max\{ f_1, \min\{ f_2, f_3 \} \}$

and the third element on the right side is computed by

$\min\{ \max\{ f_1, f_2 \}, \max\{ f_1, f_3 \} \}$.

Subcase 1: f_1 is greater than or equal to f_2 and f_3.

Then

$\max\{ f_1, \min\{ f_2, f_3 \} \} = f_1 = \min\{ f_1, f_1 \} = \min\{ \max\{ f_1, f_2 \}, \max\{ f_1, f_3 \} \}$.

Subcase 2: $f_2 \geq f_1 \geq f_3$

Then

$\max\{ f_1, \min\{ f_2, f_3 \} \} = f_1 = \min\{ f_2, f_1 \} = \min\{ \max\{ f_1, f_2 \}, \max\{ f_1, f_3 \} \}$.

Subcase 3: $f_2 \geq f_3 \geq f_1$

Then

$\max\{ f_1, \min\{ f_2, f_3 \} \} = f_3 = \min\{ f_2, f_3 \} = \min\{ \max\{ f_1, f_2 \}, \max\{ f_1, f_3 \} \}$.

Subcase 4: $f_3 \geq f_1 \geq f_2$

Then

$\max\{ f_1, \min\{ f_2, f_3 \} \} = f_1 = \min\{ f_1, f_3 \} = \min\{ \max\{ f_1, f_2 \}, \max\{ f_1, f_3 \} \}$.

Subcase 5: $f_3 \geq f_2 \geq f_1$

Then

$\max\{ f_1, \min\{ f_2, f_3 \} \} = f_2 = \min\{ f_2, f_3 \} = \min\{ \max\{ f_1, f_2 \}, \max\{ f_1, f_3 \} \}$.

All cases have been considered, so the third elements of the ordered triplets always have the same value.

Since the first and third values of the ordered triplets always have the same value, all three must always be equal and the Distributive property of $\wedge$ over $\vee$ holds.

Theorem 4.3.19: Let A, B and C be elements of NL1. The Distributive law holds for $\vee$ over $\wedge$, in other words

$$A \vee (B \wedge C) =_{NL1} (A \vee B) \wedge (A \vee C).$$

Proof: Let $A = (t_1, i_1, f_1)$, $B = (t_2, i_2, f_2)$ and $C = (t_3, i_3, f_3)$. In this case, the proof will be by examining the values of the components of the triplets generated by the operations.

Case 1: The values of the first elements in the ordered triplets.

On the left side, we would have

$$\max\{ t_1, \min\{ t_2, t_3 \} \}$$

and on the right side we would have

$$\min\{ \max\{ t_1, t_2 \}, \max \{ t_1, t_3 \} \}.$$

Subcase 1: t_1 is greater than or equal to t_2 and t_3

Then,

$$\max\{ t_1, \min\{ t_2, t_3 \} \} = t_1 = \min\{ t_1, t_1 \} = \min\{ \max\{ t_1, t_2 \}, \max \{ t_1, t_3 \} \}.$$

Subcase 2: $t_2 \geq t_1 \geq t_3$

Then,

$$\max\{ t_1, \min\{ t_2, t_3 \} \} = t_1 = \min\{ t_2, t_1 \} = \min\{ \max\{ t_1, t_2 \}, \max \{ t_1, t_3 \} \}.$$

Subcase 3: $t_2 \geq t_3 \geq t_1$

Then,

$$\max\{ t_1, \min\{ t_2, t_3 \} \} = t_3 = \min\{ t_2, t_3 \} = \min\{ \max\{ t_1, t_2 \}, \max \{ t_1, t_3 \} \}.$$

Subcase 4: $t_3 \geq t_1 \geq t_2$

Then,

$$\max\{ t_1, \min\{ t_2, t_3 \} \} = t_1 = \min\{ t_1, t_3 \} = \min\{ \max\{ t_1, t_2 \}, \max \{ t_1, t_3 \} \}.$$

Subcase 5: $t_3 \geq t_2 \geq t_1$

Then,

$\max\{ t_1, \min\{ t_2, t_3 \} \} = t_2 = \min\{ t_2, t_3 \} = \min\{ \max\{ t_1, t_2 \}, \max \{ t_1, t_3 \} \}$.

This completes the case analysis and therefore it follows that the first values of the triplets are the same in all cases.

Case 2: The values of the third elements in the ordered triplets.

On the left side, we would have

$\min\{ f_1, \max\{ f_2, f_3 \} \}$

and on the right side we have

$\max\{ \min\{ f_1, f_2 \}, \min\{ f_1, f_3 \} \}$.

Subcase 1: f_1 is less than or equal to f_2 and f_3.

Then,

$\min\{ f_1, \max\{ f_2, f_3 \} \} = f_1 = \max\{ f_1, f_1 \} = \max\{ \min\{ f_1, f_2 \}, \min\{ f_1, f_3 \} \}$.

Subcase 2: $f_2 \geq f_1 \geq f_3$

Then,

$\min\{ f_1, \max\{ f_2, f_3 \} \} = f_1 = \max\{ f_1, f_3 \} = \max\{ \min\{ f_1, f_2 \}, \min\{ f_1, f_3 \} \}$.

Subcase 3: $f_3 \geq f_1 \geq f_2$

Then,

$\min\{ f_1, \max\{ f_2, f_3 \} \} = f_1 = \max\{ f_2, f_1 \} = \max\{ \min\{ f_1, f_2 \}, \min\{ f_1, f_3 \} \}$.

Subcase 4: $f_1 \geq f_2 \geq f_3$

Then,

$\min\{ f_1, \max\{ f_2, f_3 \} \} = f_2 = \max\{ f_2, f_3 \} = \max\{ \min\{ f_1, f_2 \}, \min\{ f_1, f_3 \} \}$.

Subcase 5: $f_1 \geq f_3 \geq f_2$

Then,

$\min\{ f_1, \max\{ f_2, f_3 \} \} = f_3 = \max\{ f_2, f_3 \} = \max\{ \min\{ f_1, f_2 \}, \min\{ f_1, f_3 \} \}$.

This completes the case analysis and therefore it follows that the third values of the triplets are the same in all cases.

Therefore, the triplets are equal in all cases and $\vee$ distributes over $\wedge$.

Theorem 4.3.20: Let A, B and C be elements of NL2. The Distributive law for $\wedge$ over $\vee$ does not hold. In other words

$A \wedge (B \vee C) \neq_{NL2} (A \wedge B) \vee (A \wedge C)$.

Proof: Let A = (0.04, 0.0,0.96), B = (0.0,0.04,0.96) and C = (0.04,0.0,0.96). Then, $A \wedge B$ = (0.0, 0.04,0.96), $A \wedge C$ = (0.04,0.0,0.96) and $B \vee C$ = (0.04, 0.04, 0.92). The left side is then (0.04, 0.0, 0.96) $\wedge$ (0.04, 0.04, 0.92) = (0.0, 0.04, 0.96) and the right side is (0.0, 0.04, 0.96) $\vee$ (0.04, 0.0, 0.96) = (0.04, 0.04, 0.92).

Theorem 4.3.21: Let A, B and C be elements of NL2. The Distributive law for $\vee$ over $\wedge$ does not hold. In other words

$A \vee (B \wedge C) \neq_{NL2} (A \vee B) \wedge (A \vee C)$.

Proof: Let A = (0.0,0.03,0.97), B = (0.04, 0.0, 0.96) and C = (0.04, 0.04, 0.92). Then $B \wedge C$ = (0.0, 0.04, 0.96), $A \vee B$ = (0.04, 0.03, 0.92) and $A \vee C$ = (0.04, 0.04, 0.92). The left side is then (0.0, 0.04, 0.96) and the right side is (0.03, 0.04, 0.93).

Theorem 4.3.22: Let A, B and C be elements of INL1. Then

i) $A \wedge (B \vee C) =_{INL1} (A \wedge B) \vee (A \wedge C)$.
ii) $A \vee (B \wedge C) =_{INL1} (A \vee B) \wedge (A \vee C)$.

In other words, both distributive properties hold in INL1.

Proof:
i) Let $A = (t_1, i_1, f_1, u_1)$, $B = (t_2, i_2, f_2, u_2)$ and $C = (t_3, i_3, f_3, u_3)$. Since the u value is computed based on the values of t, i and f, if we can show the first three elements are true it will follow that the fourth elements must also be true.

$B \vee C$ = (max{t_2, t_3} , max{i_2, i_3}, min{f_2, f_3}, computed u)

$A \wedge (B \vee C)$ = (min{t_1, max{t_2, t_3}}, max{i_1, max{i_2, i_3}}, max{f_1, min{f_2, f_3}},
computed u)

$A \wedge B$ = (min{ t_1, t_2 }, max{i_1, i_2}, max{f_1, f_2}, computed u),
$A \wedge C$ = (min{ t_1, t_3 }, max{i_1, i_3}, max{f_1, f_3}, computed u),

$(A \wedge B) \vee (A \wedge C) = (\max\{\min\{ t_1, t_2 \}, \min\{ t_1, t_3 \}\},$
$\max\{\max\{i_1, i_2\}, \max\{i_1, i_3\}\},$
$\min\{\max\{f_1, f_2\}, \max\{f_1, f_3\}\},$
computed u).

Clearly, $\max\{i_1, \max\{i_2, i_3\}\} = \max\{\max\{i_1, i_2\}, \max\{i_1, i_3\}\}$ for all values, so the indeterminate terms of both 4-tuples are equal.

The remainder of the proof is a case analysis based on all possibilities.

Case 1: t_1 is smaller than t_2 and t_3. Then $t_1 = \min\{t_1, \max\{t_2, t_3\}\}$ and $t_1 = \max\{\min\{ t_1, t_2 \}, \min\{ t_1, t_3 \}\}$.

Case 2: t_1 is larger than t_2 and t_3. Then $\min\{t_1, \max\{t_2, t_3\}\}$ is the largest of t_2 and t_3 and $\max\{\min\{ t_1, t_2 \}, \min\{ t_1, t_3 \}\}$ is also the largest of t_2 and t_3.

Case 3: $t_2 \geq t_1 \geq t_3$. Then $\min\{t_1, \max\{t_2, t_3\}\} = t_1$ and $\max\{\min\{ t_1, t_2 \}, \min\{ t_1, t_3 \}\} = t_1$.

Case 4: $t_3 \geq t_1 \geq t_2$. Then $\min\{t_1, \max\{t_2, t_3\}\} = t_1$ and $\max\{\min\{ t_1, t_2 \}, \min\{ t_1, t_3 \}\} = t_1$.

Therefore, the values of the true components are equal for all possibilities.

Case 5: f_1 is larger than f_2 and f_3. Then $\max\{f_1, \min\{f_2, f_3\}\} = f_1$ and $\min\{\max\{f_1, f_2\}, \max\{f_1, f_3\}\} = f_1$.

Case 6: f_1 is smaller than f_2 and f_3. Then $\max\{f_1, \min\{f_2, f_3\}\}$ is the smallest of f_2 and f_3 and $\min\{\max\{f_1, f_2\}, \max\{f_1, f_3\}\}$ is also the smallest of f_2 and f_3.

Case 7: $f_2 \geq f_1 \geq f_3$. Then $\max\{f_1, \min\{f_2, f_3\}\} = f_1$ and $\min\{\max\{f_1, f_2\}, \max\{f_1, f_3\}\} = f_1$.

Case 8: $f_3 \geq f_1 \geq f_2$. Then $\max\{f_1, \min\{f_2, f_3\}\} = f_1$ and $\min\{\max\{f_1, f_2\}, \max\{f_1, f_3\}\} = f_1$.

Therefore, the values of the false components are equal for all possibilities.

Since the values of the first three elements of the 4-tuples are equal and the fourth is computed from the first three, it follows that they are equal.

ii) The proof is similar to that of (i) and so is omitted.

Theorem 4.3.23: Let A, B and C be elements of INL2. Then

iii) $A \wedge (B \vee C) =_{INL2} (A \wedge B) \vee (A \wedge C)$.

iv) $A \vee (B \wedge C) =_{INL2} (A \vee B) \wedge (A \vee C)$.

In other words, both distributive properties hold in INL2.

Proof: Apply a case analysis similar to that done for theorem 4.3.22.

Theorem 4.3.24: Let A, B and C be elements of PNL1. Then

i) $A \wedge (B \vee C) =_{PNL1} (A \wedge B) \vee (A \wedge C)$.
ii) $A \vee (B \wedge C) =_{PNL1} (A \vee B) \wedge (A \vee C)$.

In other words, both distributive properties hold in PNL1

Proof: Apply a case analysis similar to that done for theorem 4.3.22.

Definition 4.3.4: For many of the properties that have been examined, there has been no distinction between the $\vee$ and $\wedge$.. If a property was true, then the property with all instances of $\vee$ replaced by $\wedge$ and all instances of $\wedge$ replaced by $\vee$ was also true. When an algebra is defined with two operations that can be interchanged in this way, the two expressions are said to be **duals** of each other.

Example:
THE TWO DISTRIBUTIVE LAWS THAT WERE VERIFIED IN THE PREVIOUS THEOREMS ARE DUALS OF EACH OTHER.

Definition 4.3.5: For any set S, if there is an element $e \varepsilon S$ and an operation o such that $x \circ e = e \circ x = e$ for all $x \varepsilon S$, then e is said to be a **dominant element** for o in S.

Theorem 4.3.25: The element (0,0,1) is dominant for $\wedge$ in NL1 and there is no other element (q, r, s) in NL1 such that $(q, r, s) \wedge (t, i, f) = (q, r, s)$ for all (t, i, f) in NL1. In other words, (0,0,1) is the only dominant element in NL1.

Proof: Clearly, $(0,0,1) \wedge (t, i, f) = (0, 0, 1)$ since $\max\{1, f\} = 1$. The other two special cases of (1, 0, 0) and (0,1,0) can be eliminated as the conjunction of either with (t, i, f) where f is nonzero yields a result with a nonzero third entry.

Therefore, assume that such a (q, r, s) element exists and at least two of the entries in the triple are nonzero.

Case 1: Suppose that $0 < q < 1$. Then, there exists a value $0 < q_k < q < 1$ where $(q, r, s) \wedge (q_k, i, f)$ has a first value in the triple of q_k. By choice of q_k, we have a contradiction.

Case 2: Suppose that $0 < s < 1$. Then, there exists a value $0 < s < s_k < 1$ where $(q, r, s) \wedge (t, i, s_k)$ has a third value in the triplet of s_k. By choice of s_k, we have a contradiction.

Since at least one of the two entries examined in cases 1 and 2 must be nonzero, no such additional element can exist.

Theorem 4.3.26: The element (1, 0, 0) is dominant for $\vee$ in NL1 and there is no other element (q, r, s) in NL1 such that (q, r, s) $\vee$ (t, i, f) = (q, r, s) for all (t, i, f) in NL1. In other words, there is only one dominant element for $\vee$ in NL1.

Proof: Similar to that of theorem 4.3.25, so it is omitted.

Theorem 4.3.27:
i) (0, 1, 0) is the only dominant element in NL2 for $\wedge$.
ii) (0, 1, 0) is the only dominant element in NL2 for $\vee$.

Proof:
(a) Clearly, (0,1,0) $\wedge$ (t, i, f) = (0,1,0) for all (t,i,f) as the middle term of the result is the maximum of the middle terms. Suppose that there is another dominant element (t_1, i_1, f_1) for $\wedge$. Then $i_1 < 1.0$. However, it would then be possible to find another value i_2 such that $i_1 < i_2$ 1.0. Therefore,

(t_1,i_1, f_1) $\wedge$ (t_2, i_2, f_2)

would have a middle term of i_2, contradicting the assumption that (t_1, i_1, f_1) is a dominant term.

(b) Similar to (a) so it is omitted.

Theorem 4.3.28:
i) The element (0, 0, 1, 0) is a dominant element for $\wedge$ in INL1.
ii) The element (1, 0, 0, 0) is a dominant element for $\vee$ in INL1.
iii) The element (0, 0, 1, 0) is a dominant element for $\wedge$ in INL2.
iv) The element (1, 0, 0, 0) is a dominant element for $\vee$ in INL2.

Proof:
a) Let A = (t, i, f, u) be an element in INL1 and consider A $\wedge$ (0, 0, 1, 0). Then min {t, 0 } = 0, min{I, 0} = 0, max {f, 1} = 1 and n = 1 – 0 – 0 – 1 = 0.

b) – d) These proofs are similar to part (a) and so are omitted.

Theorem 4.3.29: Assume that there are maximum values of the t, i and f components of the triplets in PNL and call them maxt, maxi, maxf respectively. Then:
i) The element (0, maxi, maxf) is a dominant element for $\wedge$ in PNL1.
ii) The element (maxt, maxi, 0) is a dominant element for $\vee$ in PNL1.

Proof:
i) Let A = (t, i, f) be an element of PNL1 and consider A $\wedge$ (0,maxi, maxf). By definition, min{t, 0} = 0, max{0, maxi} = maxi and max{f,maxf} = maxf. Therefore, the value of the conjunction is (0, maxi, maxf).

ii) Let A = (t, i, f) be an element of PNL1 and consider A $\vee$ (maxt,maxi, 0). By definition, max{t, maxt} = maxt, max{0, maxi} = maxi and min{f,0} = 0. Therefore, the value of the disjunction is (maxt, maxi, 0).

Definition 4.3.6: Given a set S with operation $\circ$ and an identity E, if x is any element, then an element x^{-1} is the **inverse** of x if $x \circ x^{-1} = E$.

Definition 4.3.7: A monoid [S, $\circ$] with identity I is a **group** if every element x has an inverse x^{-1} such that $x \circ x^{-1} = E$.

Theorem 4.3.30:
(i) [NL1, $\vee$] is not a group.
(ii) [NL1, $\wedge$] is not a group.
(iii) [NL2, $\vee$] is not a group.
(iv) [NL2, $\wedge$] is not a group.

Proof: (i) The only property of a group that has not yet been verified is the presence of inverses. Let (t, i, f) ε NL1, where t, i, f are all not zero. The inverse with respect to $\vee$ would be an element, (t_1, i_1, f_1) such that

(t, i, f) $\vee$ (t_1, i_1, f_1) = (0, 0, 1)

since (0, 0, 1) is the identity for $\vee$. However, by definition max { t, t_1 } = 0 if and only if $t = t_1 = 0$, contradicting the general choice of t. Therefore, there is no such inverse.

(ii) Let (t, i, f) ε NL1, where t, i, f are all not zero. The inverse with respect to $\wedge$ would be an element, (t_1, i_I, f_I) such that

(t, i, f) $\wedge$ (t_1, i_I, f_I) = (1, 0, 0)

since (1, 0, 0) is the identity for $\wedge$. However, by definition, min{ t, t_1 } = 1 if and only if $t = t_1 = 1$, contradicting the general choice of t. Therefore, there is no inverse.

The proofs of (iii) and (iv) are similar and so are omitted.

Theorem 4.3.31:
i) [INL1, $\wedge$] is not a group.
ii) [INL2, $\vee$] is not a group.
iii) [INL2, $\wedge$] is not a group.
iv) [INL2, $\vee$] is not a group.
If there are identities for $\wedge$ and $\vee$ in PNL1:
v) [PNL1, $\wedge$] is not a group.
vi) [PNL1, $\vee$] is not a group.

Proof: It has already been proven that there is no identity for $\wedge$ or $\vee$ in INL1 and there is no identity for $\wedge$ or $\vee$ in INL2. Since a set must have an identity for it to be a group, results (i) – (iv) follow immediately.

v) The identity for $\wedge$ in PNL1 is (maxt, 0, 0). Take an element in PNL1 $A = (t_1, i_1, f_1)$ where i_1 is nonzero. If this element has an inverse $B = (t_2, i_2, f_2)$, then it must be true that $\max\{i_1, i_2\} = 0$. This is impossible, since i_1 is nonzero.

The proof of vi) is similar to that of v) and is omitted.

Definition 4.3.8: The classic definition of a Boolean algebra is as follows.

A Boolean algebra is a set B on which there are defined an equivalence relation "=" and two operations "+" and "*" such that the following properties hold:

(i) For all x and y in B. a + b and x * y are also in B. (Closure.)
(ii) There are elements 0 and 1 in B such that for all x in B, x + 0 = 0 and x * 1 = x. (Identity.)
(iii) For all x and y in B, x + y = y + x and x * y = y * x. (Commutativity.)
(iv) For all x, y and z in B, x + (y * z) = (x + y) * (x + z) and x * (y + z) = (x * y) + (x * z). (Distributive.)
(v) For all x in B, there is another element x' also in B, such that x + x' = 1 and x * x' = 0. (Laws of complementarity.)
(vi) There are at least two different elements in B.
(vii) If x = y, then for all z in B, x + z = y + z and x * z = y * z. (Principle of substitution.)

For our purposes, we will take NL as our set and equality to mean that

$(t, i, f) = (t_I, i_1, f_1)$

if and only if $t = t_1$, $i = i_1$ and $f = f_1$.

The operation of $\vee$ on NL1 will be the + operation and $\wedge$ the * operation. The element (0, 0, 1) is the identity for $\vee$ and (1, 0, 0) the identity for $\wedge$.

Theorem 4.3.32:
i) The algebra [NL1, $\vee$, $\wedge$] is not a Boolean algebra.
ii) The algebra [NL2, $\vee$, $\wedge$] is not a Boolean algebra.
iii) The algebra [INL1, $\vee$, $\wedge$] is not a Boolean algebra.
iv) The algebra [INL2, $\vee$, $\wedge$] is not a Boolean algebra.
v) The algebra [PNL1, $\vee$, $\wedge$] is not a Boolean algebra.

Proof:
i) It was proven in theorem 4.3.30 that there are elements in NL1 that do not have inverses. Therefore, property (v) of a Boolean algebra does not hold. In fact, the result is even stronger in that no element that is not an identity has an inverse.

The proofs of ii) through v) are identical to that of part (i).

It has been proven that property (v) of a Boolean algebra does not hold for NL2. With the exception of property (vii) of a Boolean algebra, it has been proven that NL1 satisfies all others except the distributive.

The proof of (vii) for NL1 and NL2 is rather easy and is the subject of the theorem 4.3.33.

Theorem 4.3.33:
i) NL1 satisfies the principle of substitution of a Boolean algebra. In other words, if A, B and C are elements of NL1 and B = C, then

$A \vee B =_{NL1} A \vee C$ and $A \wedge B =_{NL1} A \wedge C$.

ii) NL2 satisfies the principle of substitution of a Boolean algebra. In other words, if A, B and C are elements of NL2 and B = C, then

$A \vee B =_{NL2} A \vee C$ and $A \wedge B =_{NL2} A \wedge C$.

Proof:
i) Let $A = (t_1, i_1, f_1)$, $B = (t_2, i_2, f_2)$ and $C = (t_3, i_3, f_3)$.

$A \wedge B =$

$(t_1, i_1, f_1) \vee (t_2, i_2, f_2) = (\max\{ t_1, t_2 \},\ 1 - \max\{ t_1, t_2 \} - \min\{ f_1, f_2 \},\ \min\{ f_1, f_2 \})$

$(t_1, i_1, f_1) \vee (t_3, i_3, f_3) = (\max\{ t_1, t_3 \},\ 1 - \max\{ t_1, t_3 \} - \min\{ f_1, f_3 \},\ \min\{ f_1, f_3 \})$

If $t_2 = t_3$, $i_2 = i_3$ and $f_2 = f_3$ then

$(\max\{ t_1, t_2 \},\ 1 - \max\{ t_1, t_2 \} - \min\{ f_1, f_2 \},\ \min\{ f_1, f_2 \}) =$

$(\max\{ t_1, t_3 \},\ 1 - \max\{ t_1, t_3 \} - \min\{ f_1, f_3 \},\ \min\{ f_1, f_3 \})$

The proof for $\wedge$ is similar and so it is omitted.

ii) Similar to that of (i) and so is omitted.

Theorem 4.3.34:
i) INL1 satisfies the principle of substitution of a Boolean algebra. In other words, if A, B and C are elements of INL1 and B = C, then

$A \vee B =_{INL1} A \vee C$ and $A \wedge B =_{INL1} A \wedge C$.

ii) INL2 satisfies the principle of substitution of a Boolean algebra. In other words, if A, B and C are elements of INL2 and B = C, then

$A \vee B =_{INL2} A \vee C$ and $A \wedge B =_{INL2} A \wedge C$.

iii) PNL1 satisfies the principle of substitution of a Boolean algebra. In other words, if A, B and C are elements of PNL1 and B = C, then

$A \vee B =_{PNL1} A \vee C$ and $A \wedge B =_{PNL1} A \wedge C$.

Proof: The proof of each result is similar to that of theorem 4.3.33.

Therefore, with the exception of the inverse properties, NL1 satisfies all of the properties of a Boolean algebra. Since it is so close to a Boolean algebra, NL1 satisfies several other properties commonly associated with a Boolean algebra. NL2 is farther from a Boolean algebra, as it also does not satisfy the distributive properties. INL1, INL2 and PNL1 are also very similar to a Boolean algebra.

Theorem 4.3.35: NL1 satisfies DeMorgan's rules of the distribution of the negation. In other words, if A and B are elements of NL1, then

(i) $\neg (A \vee B) =_{NL1} \neg A \wedge \neg B$

(ii) $\neg (A \wedge B) =_{NL1} \neg A \vee \neg B$.

Proof: Let $A = (t_1, i_1, f_1)$ and $B = (t_2, i_2, f_2)$.

(i) $\neg (A \vee B) = \neg (\max\{ t_1, t_2 \}, 1 - \max\{ t_1, t_2 \} - \min\{ f_1, f_2 \}, \min\{ f_1, f_2 \}) =$

$(\min\{ f_1, f_2 \}, 1 - \max\{ t_1, t_2 \} - \min\{ f_1, f_2 \}, \max\{ t_1, t_2 \}) = (f_1, i_1, t_1) \wedge (f_2, i_2, t_2) =$

$\neg (t_1, i_1, f_1) \wedge \neg(t_2, i_2, f_2) = \neg A \wedge \neg B$.

(ii) $\neg (A \wedge B) = \neg (\min\{ t_1, t_2 \}, 1 - \min\{ t_1, t_2 \} - \max\{ f_1, f_2 \}, \max\{ f_1, f_2 \}) =$

$(\max\{ f_1, f_2 \}, 1 - \min\{ t_1, t_2 \} - \max\{ f_1, f_2 \}, \min\{ t_1, t_2 \}) = (f_1, i_1, t_1) \vee (f_2, i_2, t_2) =$

$\neg (t_1, i_1, f_1) \vee \neg(t_2, i_2, f_2) = \neg A \vee \neg B$.

Theorem 4.3.36: NL2 satisfies DeMorgan's rules of the distribution of the negation. In other words, if A and B are elements of NL1, then

i) $\neg (A \vee B) =_{NL2} \neg A \wedge \neg B$.

ii) $\neg (A \wedge B) =_{NL2} \neg A \vee \neg B$.

Proof: Let $A = (t_1, i_1, f_1)$ and $B = (t_2, i_2, f_2)$.

i) $A \vee B = (\min \{ 1 - \max \{i_1, i_2 \}, \max \{t_1, t_2 \} \}, \max \{i_1, i_2 \},$
$1 - \max \{i_1, i_2 \} - \min \{ 1 - \max \{i_1, i_2 \}, \max \{t_1, t_2 \} \})$

$\neg (A \vee B) = (1 - \max \{i_1, i_2 \} - \min \{ 1 - \max \{i_1, i_2 \}, \max \{t_1, t_2 \} \}, \max \{i_1, i_2 \},$
$\min \{ 1 - \max \{i_1, i_2 \}, \max \{t_1, t_2 \} \})$

$\neg A = (f_1, i_1, t_1) \quad \neg B = (f_2, i_2, t_2)$

$\neg A \wedge \neg B = (1 - \max \{i_1, i_2 \} - \min \{ 1 - \max \{i_1, i_2 \}, \max \{t_1, t_2 \}\}, \max \{i_1, i_2 \},$
$\min \{ 1 - \max \{i_1, i_2 \}, \max \{t_1, t_2 \}\})$

ii) The proof of (ii) is similar and so is omitted.

Theorem 4.3.37: INL1 satisfies DeMorgan's rules of the distribution of the negation. In other words, if A and B are elements of INL1, then

i) $\neg (A \vee B) =_{INL1} \neg A \wedge \neg B.$

ii) $\neg (A \wedge B) =_{INL1} \neg A \vee \neg B.$

Proof: Let $A = (t_1, i_1, f_1, u_1)$ and $B = (t_2, i_2, f_2, u_2)$.

i) Then, $A \vee B = (\max \{t_1, t_2 \}, \min\{i_1, i_2 \}, \min \{f_1, f_2 \},$ computed u) and

$\neg(A \vee B) = (\min \{f_1, f_2 \}, \min\{i_1, i_2 \}, \max\{t_1, t_2 \},$ computed u).

$\neg A = (f_1, i_1, t_1, u_1)$, $\neg B = (f_2, i_2, t_2, u_2)$ and

$\neg A \wedge \neg B = (\min \{f_1, f_2 \}, \min\{i_1, i_2 \}, \max\{t_1, t_2 \},$ computed u).

ii) The proof is similar and so is omitted.

Theorem 4.3.38: INL2 satisfies DeMorgan's rules of the distribution of the negation. In other words, if A and B are elements of INL2, then

i) $\neg (A \vee B) =_{INL2} \neg A \wedge \neg B.$

ii) $\neg (A \wedge B) =_{INL2} \neg A \vee \neg B.$

Proof: Almost identical to that of theorem 4.3.37.

Theorem 4.3.39: PNL1 satisfies DeMorgan's rules of the distribution of the negation. In other words, if A and B are elements of PNL1, then

i) $\neg (A \vee B) =_{PNL1} \neg A \wedge \neg B.$

ii) $\neg (A \wedge B) =_{PNL1} \neg A \vee \neg B$.

Proof: Similar to that of theorem 4.3.37.

Definition 4.3.9: An expression in NL1 is said to be well-formed if it satisfies the following properties:

(i) (t, i, f) where t + i + f = 1.0 is well-formed.
(ii) The constants T = (1, 0, 0), I = (0, 1, 0) and F = (0, 0, 1) are well-formed.
(iii) If A is well-formed in NL, then so is ¬ A.
(iv) If A is well-formed, then so is (A).
(v) If A and B are well-formed in NL, then so are A ∧ B and A ∨ B.
(vi) Nothing that cannot be formed using rules {i) – (v) a finite number of times is well-formed.

A similar definition can be used for NL2, INL1, INL2 and PNL1.

Theorem 4.3.40:
i) If a well-formed expression A in NL1 is an absolute tautology, then A contains the (1, 0, 0) element.
ii) If a well-formed expression A in NL1 is an absolute contradiction, then A contains the (0, 0, 1) element.

Proof: As always, the expression A is constructed using the three connectives, { ¬, ∨, ∧ }. The proof of both parts will be combined into one, as the results are similar.

We start the proof with a lemma.

Lemma: If A and B are elements of NL1, then:

i) If A ∨ B = (1, 0, 0) then either A or B is (1, 0, 0).
ii) If A ∧ B = (1, 0, 0) then either A or B is (1, 0, 0).
iii) If A ∨ B = (0, 0, 1) then either A or B is (0, 0, 1).
iv) If A ∧ B = (0, 0, 1) then either A or B is (0, 0, 1).
v) If ¬A = (1, 0, 0) then A = (0, 0, 1).
vi) If ¬A = (0, 0, 1) then A = (1, 0, 0).

Proof of lemma: Results i) through iv) are consequences of the definitions of ∨ and ∧ and v) and vi) are consequences of the definition of the negation.

Proof of theorem:
Since A is well-formed in NL1, it is constructed by using a finite number of the symbols in { ¬, ∧, ∨ }. Therefore, the proof will be by induction on the number of operations used to construct the expression.

Basis step: Clearly if the expression A is a triplet, then A is an absolute tautology if and only if A = (1, 0, 0) and an absolute contradiction if and only if A = (0, 0, 1).

Inductive step: Assume that for any expression A constructed using k or fewer steps, if A is an absolute tautology, then (1, 0, 0) appears in A. Also assume that if A is constructed in the same manner and is an absolute contradiction, then (0, 0, 1) appears in A.

Let A and B be any expressions satisfying the inductive assumptions. There are four rules that can be applied.

Case 1: Rule iii) of the definition of a wff is applied to A.
The only value for A that can yield (1, 0, 0) is (0, 0, 1) and the only value that can yield (0, 0, 1) is (1, 0, 0). Therefore, the only way that ¬A can be an absolute tautology is if A is an absolute contradiction and the only way that ¬A can be an absolute contradiction is if A is an absolute tautology. Applying the induction hypotheses, it follows that if ¬A is an absolute tautology or contradiction it is because A contains (1, 0, 0) or (0, 0, 1).

Case 2: Rule iv) of the definition of a wff is applied to A.
Since enclosing the expression in parentheses does nothing to the value of the expression, if follows that (A) is an absolute tautology or contradiction because A contains either (1, 0, 0) or (0, 0, 1).

Case 3: Rule v) of the definition of a wff is applied to form A ∧ B.
From the lemma, we have that if this expression is an absolute tautology, then either A or B is. We can then apply the inductive hypothesis to conclude that if A ∧ B is an absolute tautology, then it contains the element (1, 0, 0). Similar reasoning will yield the comparable result for A ∧ B being an absolute contradiction.

Case 4: Rule v) of the definition of a wff is applied to form A ∨ B.
From the lemma, we have that if this expression is an absolute tautology, then either A or B is. We can then apply the inductive hypothesis to conclude that if A ∨ B is an absolute tautology, then it contains the element (1, 0, 0). Similar reasoning will yield the comparable result for A ∨ B being an absolute contradiction.

Therefore, by the principle of mathematical induction, we have reached the desired conclusion.

The theorem for NL2 is similar, but has an additional restriction.

Theorem 4.3.41:

i) If a well-formed expression A in NL2 is an absolute tautology, then A contains the (1, 0, 0) element and all middle terms of the triplets in A are zero.

ii) If a well-formed expression A in NL2 is an absolute contradiction, then A contains the (0, 0, 1) element and all middle terms of the triplets in A are zero.

Again, we start the proof with a lemma.

Lemma: If A and B are elements of NL2, then:

i) If A ∨ B = (1, 0, 0) then either A or B is (1, 0, 0) and the middle terms in the triplets A and B are zero.
ii) If A ∧ B = (1, 0, 0) then either A or B is (1, 0, 0) and the middle terms in the triplets A and B are zero..
iii) If A ∨ B = (0, 0, 1) then either A or B is (0, 0, 1) and the middle terms in the triplets A and B are zero.
iv) If A ∧ B = (0, 0, 1) then either A or B is (0, 0, 1) and the middle terms in the triplets A and B are zero.
v) If ¬A = (1, 0, 0) then A = (0, 0, 1).
vi) If ¬A = (0, 0, 1) then A = (1, 0, 0).

Proof of lemma:
(i) If either A or B had a middle term that was not zero, then by the definition of ∨ the middle term of the expression must be nonzero. With the middle terms zero, the first term is the maximum of the two first terms and the result follows.
(iii) – (vi) The proofs are similar to that of (i) and is omitted.

Proof of the theorem:
Similar to that of theorem 4.3.40.

Theorem 4.3.42:
i) If A is an element of INL1 and A = (1, 0, 0, 0), then A contains the (1,0,0,0) element.
ii) If A is an element of INL1 and A = (0,0,1,0), then A contains the (0,0,1,0) element.
iii) If A is an element of INL2 and A = (1,0,0,0), then A contains the (1,0,0,0) element.
iv) If A is an element of INL2 and A = (0,0,1,0) then A contains the (0,0,1,0) element.

Proof: Similar to that of theorem 4.3.40.

Section 4
Defining Other Connectives in Neutrosophic Logic

As was mentioned in the sections on the classical, three-valued and fuzzy logics, it is possible to define additional connectives as abbreviations for expressions built using { ¬, ∧, ∨ }. However, it is difficult to define logical equality when using real numbers as a difference due to rounding error would lead to a conclusion of inequality. A similar problem occurs when using logical inequality. Therefore, if these connectives are to be defined in a useful manner, there must be some error term that can be used to define a

region, “close enough to be considered equal.” For these reasons, we will not define NL1 or NL2 equivalents of logical equivalence or exclusive or.

Definition 4.4.1: If A and B are elements of NL1, then the implication $A \rightarrow_{NL1} B$ is an abbreviation for $\neg A \vee B$.

Example:
If A = (.12, .24, .64) and B = (.45, .06, .49) then

$A \rightarrow_{NL1} B = (.64, .24, .12) \vee (.45, .06, .49) = (.64, .24, .12).$

Definition 4.4.2: If A and B are elements of NL2, then the implication $A \rightarrow_{NL2} B$ is an abbreviation for $\neg A \vee B$.

Example:
If A = (.12, .24, .64) and B = (.45, .06, .49) then

$A \rightarrow_{NL2} B = (.64, .24, .12) \vee (.45, .06, .49) = (.64, .24, .12).$

Definition 4.4.3: If A and B are elements of INL1, then the implication $A \rightarrow_{INL1} B$ is an abbreviation for $\neg A \vee B$.

Example:
If A = (.12, .24, .34, .30) and B = (.45, .06, .29, .20) then

$A \rightarrow_{INL1} B = (.34, .24, .12, .30) \vee (.45, .06, .29, .20) = (.45, .06, .12, .37).$

Definition 4.4.4: If A and B are elements of INL2, then the implication $A \rightarrow_{INL2} B$ is an abbreviation for $\neg A \vee B$.

Example:
If A = (.12, .24, .34, .30) and B = (.45, .06, .29, .20) then

$A \rightarrow_{INL2} B = (.34, .24, .12, .30) \vee (.45, .06, .29, .20) = (.45, .23, .12, .20).$

Definition 4.4.5: If A and B are elements of PNL1, then the implication $A \rightarrow_{PNL1} B$ is an abbreviation for $\neg A \vee B$.

Example:
If A = (.12, .24, .74) and B = (.42, .06, .59) then

$A \rightarrow_{PNL1} B = (.74, .24, .12) \vee (.45, .06, .59) = (.74, .24, .12).$

Definition 4.4.6: If A and B are elements of NL1, then the joint denial $\downarrow_{NL1}$ is an abbreviation for $\neg(A \vee B)$.

Example:
If A = (.12, .24, .64) and B = (.45, .06, .49) then

$A \downarrow_{NL1} B = \neg(A \vee B) = \neg(.45, .06, .49) = (.49, .06, .45).$

Definition 4.4.7: If A and B are elements of NL2, then the joint denial $\downarrow_{NL2}$ is an abbreviation for $\neg(A \vee B)$.

Example:
If A = (.12, .24, .64) and B = (.45, .06, .49) then

$A \downarrow_{NL2} B = \neg(A \vee B) = \neg(.45, .24, .31) = (.31, .24, .45).$

Definition 4.4.8: If A and B are elements of INL1, then the joint denial $\downarrow_{INL1}$ is an abbreviation for $\neg(A \vee B)$.

Example:
If A = (.12, .24, .34, .30) and B = (.45, .06, .29, .20) then

$A \downarrow_{INL1} B = \neg(A \vee B) = \neg(.45, .06, .29, .30) = (.29, .06, .45, .30).$

Definition 4.4.9: If A and B are elements of INL2, then the joint denial $\downarrow_{INL2}$ is an abbreviation for $\neg(A \vee B)$.

Example:
If A = (.12, .24, .34, .30) and B = (.45, .06, .29, .20) then

$A \downarrow_{INL2} B = \neg(A \vee B) = \neg(.45, .06, .29, .20) = (.29, .06, .45, .20).$

Definition 4.4.10: If A and B are elements of PNL1, then the joint denial $\downarrow_{PNL1}$ is an abbreviation for $\neg(A \vee B)$.

Example:
If A = (.12, .24, .74) and B = (.42, .06, .59) then

$A \downarrow_{PNL1} B = \neg(A \vee B) = \neg(.42, .24, .59) = (.59, .24, .42).$

Definition 4.4.11: If A and B are elements of NL1, then the alternative denial $|_{NL1}$ is an abbreviation for $\neg(A \wedge B)$.

Example:
If A = (.12, .24, .64) and B = (.45, .06, .49) then

$A |_{NL1} B = \neg(A \wedge B) = \neg(.12, .24, .64) = (.64, .24, .12).$

Definition 4.4.12: If A and B are elements of NL2, then the alternative denial $|_{NL2}$ is an abbreviation for ¬(A ∧ B).

Example:
If A = (.12, .24, .64) and B = (.45, .06, .49) then

A $|_{NL2}$ B = ¬(A ∧ B) = ¬(.12, .24, .64) = (.64, .24, .12).

Definition 4.4.13: If A and B are elements of INL1, then the alternative denial $|_{INL1}$ is an abbreviation for ¬(A ∧ B).

Example:
If A = (.12, .24, .34, .30) and B = (.45, .06, .29, .20) then

A $|_{INL1}$ B = ¬(A ∧ B) = ¬(.12, .06, .34, .48) = (.34, .06, .12, .48).

Definition 4.4.14: If A and B are elements of INL2, then the alternative denial $|_{INL2}$ is an abbreviation for ¬(A ∧ B).

Example:
If A = (.12, .24, .34, .30) and B = (.45, .06, .29, .20) then

A $|_{INL2}$ B = ¬(A ∧ B) = ¬(.12, .34 , .34, .20) = (..34, .34, .12, .20).

Definition 4.4.15: If A and B are elements of PNL1, then the alternative denial $|_{PNL1}$ is an abbreviation for ¬(A ∧ B).

Example:
If A = (.12, .24, .74) and B = (.42, .06, .59) then

A $|_{PNL1}$ B = ¬(A ∧ B) = ¬(.12, .24, .74) = (.74, .24, .12).

Theorem 4.4.1: The joint and alternative denial connectives for NL1, NL2, INL1, INL2 and PNL1 are commutative. However, the implication is not.

Proof: Since the joint denial (A↓B) is an abbreviation for ¬(A ∨ B) and the alternative denial (A|B) is an abbreviation for ¬(A ∧ B) and both ∨ and ∧ are commutative the result follows. However, the implication (A → B) is an abbreviation for (¬A ∨ B) which is not the same as (¬B ∨ A).

If these three additional connectives are to be used in NL1, NL2, INL1, INL2 and PNL1 then rule v) of the definition of a well-formed expression could be modified to

v) If A and B are well-formed in NL, then so are A ∧ B, A ∨ B, A → B, A ↓ B and A | B.

Section 5
Implementing the Neutrosophic Connectives in Computer Programs

The following Java class is an implementation of the elements of NL1.

```
/* This class is an implementation of the elements of the Neutrosophic Logic (NL). It was
developed by Charles Ashbacher 12/31/2000. */

public class NLElement
{

 private float truthvalue;
 private float indeterminatevalue;
 private float falsevalue;

/* This function is called when a new instance is created. The values are checked for
 conformance to the rule that the values must sum to 1.0. Since floating point addition is
not precise, the test allows for some inaccuracy. */

 public NLElement(float tvalue, float ivalue, float fvalue)
 {
  float test;
  test=Math.abs(1.0f - (tvalue+ivalue+fvalue));
  if(test<0.00001f)
  {
   truthvalue=tvalue;
   indeterminatevalue=ivalue;
   falsevalue=fvalue;
  }
  else
  {
   System.out.println("The inputs do not satisfy the criteria that they sum to 1.0");
   System.out.println("The object has been set to indeterminate");
   truthvalue=falsevalue=0.0f;
   indeterminatevalue=1.0f;
  }
 }

/* This function is used to update the contents of an NL element. The values are checked
for conformance to the rule that the values must sum to 1.0. Since floating point addition
 is not precise, the test allows for some inaccuracy. */

 public void updatevalues(float tvalue, float ivalue, float fvalue)
```

```
{
 float test;
 test=Math.abs(1.0f - (tvalue+ivalue+fvalue));
 if(test<0.00001f)
 {
  truthvalue=tvalue;
  indeterminatevalue=ivalue;
  falsevalue=fvalue;
 }
 else
 {
  System.out.println("The inputs do not satisfy the criteria that they sum to 1.0");
  System.out.println("The object has been set to indeterminate");
  truthvalue=falsevalue=0.0f;
  indeterminatevalue=1.0f;
 }
}

public float gettrue()
{
 return truthvalue;
}

public float getindeterminate()
{
 return indeterminatevalue;
}

public float getfalse()
{
 return falsevalue;
}

public void printvalues()
{
 System.out.println("The truth value is "+truthvalue);
 System.out.println("The indeterminate value is "+indeterminatevalue);
 System.out.println("The false value is "+falsevalue);
}

}
```

The following Java program contains functions that implement the ∧, ∨, ¬, →, ↓ and | connectives of the NL1 logic.

```
/* This program implements the connectives defined in the NL1 logic. It was written by
Charles Ashbacher 12/31/2000. The names of the function should be self-explanatory as
to which connective it implements. */

public class UsingNLElement
{
  public static NLElement NLand(NLElement nle1, NLElement nle2)
  {
   float nle1true,nle1false;
   float nle2true,nle2false;
   float assigntrue,assignfalse;
   nle1true=nle1.gettrue();
   nle1false=nle1.getfalse();
   nle2true=nle2.gettrue();
   nle2false=nle2.getfalse();
   if(nle1true>=nle2true)
   {
    assigntrue=nle2true;
   }
   else
   {
    assigntrue=nle1true;
   }
   if(nle1false>=nle2false)
   {
    assignfalse=nle1false;
   }
   else
   {
    assignfalse=nle2false;
   }
   NLElement tempNL=new NLElement(assigntrue,
                                  1.0f-(assigntrue+assignfalse),assignfalse);
   return tempNL;
  }

  public static NLElement NLor(NLElement nle1, NLElement nle2)
  {
   float nle1true,nle1false;
   float nle2true,nle2false;
   float assigntrue,assignfalse;
   nle1true=nle1.gettrue();
   nle1false=nle1.getfalse();
   nle2true=nle2.gettrue();
   nle2false=nle2.getfalse();
   if(nle1true>=nle2true)
```

```
 {
  assigntrue=nle1true;
 }
 else
 {
  assigntrue=nle2true;
 }
 if(nle1false>=nle2false)
 {
  assignfalse=nle2false;
 }
 else
 {
  assignfalse=nle1false;
 }
 NLElement tempNL=new NLElement(assigntrue,
                    1.0f-(assigntrue+assignfalse),assignfalse);
 return tempNL;
}

public static NLElement NLnegation(NLElement nle1)
{
 float assigntrue=nle1.getfalse();
 float assignfalse=nle1.gettrue();
 NLElement tempNL=new NLElement(assigntrue,
                  1.0f-(assigntrue+assignfalse),assignfalse);
 return tempNL;
}

public static NLElement NLimplication(NLElement nle1,NLElement nle2)
{
 NLElement nle3=NLnegation(nle1);
 NLElement nle4=NLor(nle3,nle2);
 return(nle4);
}

public static NLElement NLjointdenial(NLElement nle1,NLElement nle2)
{
 NLElement nle3=NLor(nle1,nle2);
 NLElement nle4=NLnegation(nle3);
 return (nle4);
}

public static NLElement NLaltdenial(NLElement nle1,NLElement nle2)
{
 NLElement nle3=NLand(nle1,nle2);
```

```
 NLElement nle4=NLnegation(nle3);
 return (nle4);
}

public static void main(String args[])
{
 NLElement nle1=new NLElement(0.2f,0.3f,0.5f);
 NLElement nle2=new NLElement(0.1f,0.3f,0.6f);
 NLElement nle3;
 nle3=NLand(nle1,nle2);
 System.out.println("The original values are ");
 nle1.printvalues();
 nle2.printvalues();
 System.out.println("The result of NL and is ");
 nle3.printvalues();
 nle3=NLor(nle1,nle2);
 System.out.println("The result of NL or is ");
 nle3.printvalues();
 nle3=NLnegation(nle1);
 System.out.println("The result of NL negation is ");
 nle3.printvalues();
 nle3=NLimplication(nle1,nle2);
 System.out.println("The result of NL implication is ");
 nle3.printvalues();
 nle3=NLjointdenial(nle1,nle2);
 System.out.println("The result of NL joint denial is ");
 nle3.printvalues();
 nle3=NLaltdenial(nle1,nle2);
 System.out.println("The result of NL alternative denial is ");
 nle3.printvalues();
 }
}
```

The output in this case is

```
The original values are
The truth value is .2
The indeterminate value is .3
The false value is .5
The truth value is .1
The indeterminate value is .3
The false value is .6
The result of NL and is
The truth value is .1
The indeterminate value is .3
The false value is .6
The result of NL or is
```

```
The truth value is .2
The indeterminate value is .3
The false value is .5
The result of NL negation is
The truth value is .5
The indeterminate value is .3
The false value is .2
The result of NL implication is
The truth value is .5
The indeterminate value is .3
The false value is .2
The result of NL joint denial is
The truth value is .5
The indeterminate value is .3
The false value is .2
The result of NL alternative denial is
The truth value is .6
The indeterminate value is .3
The false value is .1
```

The following Java code implements the elements of Intuitionistic Neutrosophic Logic.

```
/* Written by Charles Ashbacher June, 2002. */

public class INLElement
{
 private float truthvalue;
 private float indeterminatevalue;
 private float falsevalue;
 private float unknownvalue;

 public INLElement(float tvalue, float ivalue, float fvalue, float uvalue)
 {
  float test;
  test=Math.abs(1.0f - (tvalue+ivalue+fvalue+uvalue));
  if(test<0.00001f)
  {
   truthvalue=tvalue;
   indeterminatevalue=ivalue;
   falsevalue=fvalue;
   unknownvalue=uvalue;
  }
  else
  {
   System.out.println("The inputs do not satisfy the criteria that they sum to 1.0");
   System.out.println("The object has been set to unknown");
   truthvalue=falsevalue= indeterminatevalue=0.0f;
```

```
  unknownvalue=1.0f;
 }
}

public void updatevalues(float tvalue, float ivalue, float fvalue, float uvalue)
{
 float test;
 test=Math.abs(1.0f - (tvalue+ivalue+fvalue+uvalue));
 if(test<0.00001f)
 {
  truthvalue=tvalue;
  indeterminatevalue=ivalue;
  falsevalue=fvalue;
  unknownvalue=uvalue;
 }
 else
 {
  System.out.println("The inputs do not satisfy the criteria that they sum to 1.0");
  System.out.println("The object has been set to unknown");
  truthvalue=falsevalue=  indeterminatevalue=0.0f;
  unknownvalue=1.0f;
 }
}

public float gettrue()
{
 return truthvalue;
}

public float getindeterminate()
{
 return indeterminatevalue;
}

public float getfalse()
{
 return falsevalue;
}

public float getunknown()
{
 return unknownvalue;
}

public void printvalues()
{
```

```
  System.out.println("The truth value is "+truthvalue);
  System.out.println("The indeterminate value is "+indeterminatevalue);
  System.out.println("The false value is "+falsevalue);
  System.out.println("The unknown value is "+unknownvalue);
  System.out.println(" ");
 }

}
```

The following Java program contains functions that implement the ∧, ∨, ¬, →, ↓ and | connectives of the NL1 logic.

```
/* Written by Charles Ashbacher, June, 2002. */

public class UsingINLElement
{
  public static INLElement INL1and(INLElement nle1, INLElement nle2)
  {
   float nle1true,nle1false;
   float nle2true,nle2false;
   float assigntrue,assignfalse,assignindeterminate,assignunknown;
   float nle1indeterminate,nle2indeterminate;

// Get the values of the components needed in the computation
   nle1true=nle1.gettrue();
   nle1false=nle1.getfalse();
   nle1indeterminate=nle1.getindeterminate();
   nle2true=nle2.gettrue();
   nle2false=nle2.getfalse();
   nle2indeterminate=nle2.getindeterminate();

// Compute the and
   assigntrue=getMin(nle1true,nle2true);
   assignfalse=getMax(nle1false,nle2false);
   assignindeterminate=getMin(nle1indeterminate,nle2indeterminate);
   assignunknown=1.0f - assigntrue-assignfalse-assignindeterminate;
   INLElement tempNL=new
INLElement(assigntrue,assignindeterminate,assignfalse,assignunknown);
   return tempNL;
  }

  public static INLElement INL2and(INLElement nle1, INLElement nle2)
  {
   float nle1true,nle1false;
   float nle2true,nle2false;
   float assigntrue,assignfalse,assignindeterminate,assignunknown;
```

```
   float nle1unknown,nle2unknown;

// Get the values of the components needed in the computation
   nle1true=nle1.gettrue();
   nle1false=nle1.getfalse();
   nle1unknown=nle1.getunknown();
   nle2true=nle2.gettrue();
   nle2false=nle2.getfalse();
   nle2unknown=nle2.getunknown();

// Compute the and
   assigntrue=getMin(nle1true,nle2true);
   assignfalse=getMax(nle1false,nle2false);
   assignunknown=getMin(nle1unknown,nle2unknown);
   assignindeterminate=1.0f - assigntrue-assignfalse-assignunknown;
   INLElement tempNL=new
INLElement(assigntrue,assignindeterminate,assignfalse,assignunknown);
   return tempNL;
  }

  public static INLElement INL1or(INLElement nle1, INLElement nle2)
  {
   float nle1true,nle1false;
   float nle2true,nle2false;
   float nle1indeterminate,nle2indeterminate;
   float assigntrue,assignfalse,assignindeterminate,assignunknown;

// Get the values to be used in the computation
   nle1true=nle1.gettrue();
   nle1false=nle1.getfalse();
   nle1indeterminate=nle1.getindeterminate();
   nle2true=nle2.gettrue();
   nle2false=nle2.getfalse();
   nle2indeterminate=nle2.getindeterminate();

// Compute the or
   assigntrue=getMax(nle1true,nle2true);
   assignfalse=getMin(nle1false,nle2false);
   assignindeterminate=getMin(nle1indeterminate,nle2indeterminate);
   assignunknown=1.0f - assigntrue-assignfalse-assignindeterminate;
   INLElement tempNL=new
INLElement(assigntrue,assignindeterminate,assignfalse,assignunknown);
   return tempNL;
  }

  public static INLElement INL2or(INLElement nle1, INLElement nle2)
```

```
  {
   float nle1true,nle1false;
   float nle2true,nle2false;
   float nle1unknown,nle2unknown;
   float assigntrue,assignfalse,assignindeterminate,assignunknown;

// Get the values to be used in the computation
   nle1true=nle1.gettrue();
   nle1false=nle1.getfalse();
   nle1unknown=nle1.getunknown();
   nle2true=nle2.gettrue();
   nle2false=nle2.getfalse();
   nle2unknown=nle2.getunknown();

// Compute the or
   assigntrue=getMax(nle1true,nle2true);
   assignfalse=getMin(nle1false,nle2false);
   assignunknown=getMin(nle1unknown,nle2unknown);
   assignindeterminate=1.0f - assigntrue-assignfalse-assignunknown;
   INLElement tempNL=new
INLElement(assigntrue,assignindeterminate,assignfalse,assignunknown);
   return tempNL;
  }

// The definition of the negation is the same for INL1 and INL2, so
// there is only one negation operation.
 public static INLElement INLnegation(INLElement nle1)
 {
  float assigntrue=nle1.getfalse();
  float assignfalse=nle1.gettrue();
  float assignindeterminate=nle1.getindeterminate();
  float assignunknown=nle1.getunknown();
   INLElement tempNL=new
INLElement(assigntrue,assignindeterminate,assignfalse,assignunknown);
  return tempNL;
 }

 public static INLElement INL1implication(INLElement nle1,INLElement nle2)
 {
  INLElement nle3=INLnegation(nle1);
  INLElement nle4=INL1or(nle3,nle2);
  return(nle4);
 }

 public static INLElement INL2implication(INLElement nle1,INLElement nle2)
 {
```

```
 INLElement nle3=INLnegation(nle1);
 INLElement nle4=INL2or(nle3,nle2);
 return(nle4);
}

public static INLElement INL1jointdenial(INLElement nle1,INLElement nle2)
{
 INLElement nle3=INL1or(nle1,nle2);
 INLElement nle4=INLnegation(nle3);
 return (nle4);
}

public static INLElement INL2jointdenial(INLElement nle1,INLElement nle2)
{
 INLElement nle3=INL2or(nle1,nle2);
 INLElement nle4=INLnegation(nle3);
 return (nle4);
}

public static INLElement INL1altdenial(INLElement nle1,INLElement nle2)
{
 INLElement nle3=INL1and(nle1,nle2);
 INLElement nle4=INLnegation(nle3);
 return (nle4);
}

public static INLElement INL2altdenial(INLElement nle1,INLElement nle2)
{
 INLElement nle3=INL2and(nle1,nle2);
 INLElement nle4=INLnegation(nle3);
 return (nle4);
}

public static float getMin(float x1,float x2)
{
 float theMin;
 if(x1>=x2)
 {
  theMin=x2;
 }
 else
 {
  theMin=x1;
 }
 return theMin;
}
```

```
public static float getMax(float x1,float x2)
{
 float theMax;
 if(x1>=x2)
 {
  theMax=x1;
 }
 else
 {
  theMax=x2;
 }
 return theMax;
}

public static void main(String args[])
{
 INLElement nle1=new INLElement(0.3f,0.1f,0.5f,0.1F);
 INLElement nle2=new INLElement(0.1f,0.3f,0.2f,0.4f);
 INLElement nle3;
 nle3=INL1and(nle1,nle2);
 System.out.println("The original values are ");
 nle1.printvalues();
 nle2.printvalues();
 System.out.println("The result of NL and is ");
 nle3.printvalues();
 nle3=INL1or(nle1,nle2);
 System.out.println("The result of NL or is ");
 nle3.printvalues();
 nle3=INLnegation(nle1);
 System.out.println("The result of NL negation is ");
 nle3.printvalues();
 nle3=INL1implication(nle1,nle2);
 System.out.println("The result of NL implication is ");
 nle3.printvalues();
 nle3=INL1jointdenial(nle1,nle2);
 System.out.println("The result of NL joint denial is ");
 nle3.printvalues();
 nle3=INL1altdenial(nle1,nle2);
 System.out.println("The result of NL alternative denial is ");
 nle3.printvalues();
 }
}
```

Notice that there are separate operations for the connectives in INL1 and INL2. The output when this program is run, where the only connectives used are those for INL1, is as follows.

```
The original values are
The truth value is 0.3
The indeterminate value is 0.1
The false value is 0.5
The unknown value is 0.1

The truth value is 0.1
The indeterminate value is 0.3
The false value is 0.2
The unknown value is 0.4

The result of NL and is
The truth value is 0.1
The indeterminate value is 0.1
The false value is 0.5
The unknown value is 0.29999998

The result of NL or is
The truth value is 0.3
The indeterminate value is 0.1
The false value is 0.2
The unknown value is 0.4

The result of NL negation is
The truth value is 0.5
The indeterminate value is 0.1
The false value is 0.3
The unknown value is 0.1

The result of NL implication is
The truth value is 0.5
The indeterminate value is 0.1
The false value is 0.2
The unknown value is 0.20000002

The result of NL joint denial is
The truth value is 0.2
The indeterminate value is 0.1
The false value is 0.3
The unknown value is 0.4

The result of NL alternative denial is
The truth value is 0.5
```

```
The indeterminate value is 0.1
The false value is 0.1
The unknown value is 0.29999998
```

If the main function is changed to the following, where the connectives for INL2 are used,

```
public static void main(String args[])
{
 INLElement nle1=new INLElement(0.3f,0.1f,0.5f,0.1F);
 INLElement nle2=new INLElement(0.1f,0.3f,0.2f,0.4f);
 INLElement nle3;
 nle3=INL2and(nle1,nle2);
 System.out.println("The original values are ");
 nle1.printvalues();
 nle2.printvalues();
 System.out.println("The result of NL and is ");
 nle3.printvalues();
 nle3=INL2or(nle1,nle2);
 System.out.println("The result of NL or is ");
 nle3.printvalues();
 nle3=INLnegation(nle1);
 System.out.println("The result of NL negation is ");
 nle3.printvalues();
 nle3=INL2implication(nle1,nle2);
 System.out.println("The result of NL implication is ");
 nle3.printvalues();
 nle3=INL2jointdenial(nle1,nle2);
 System.out.println("The result of NL joint denial is ");
 nle3.printvalues();
 nle3=INL2altdenial(nle1,nle2);
 System.out.println("The result of NL alternative denial is ");
 nle3.printvalues();
}
```

The output is as follows

```
The original values are
The truth value is 0.3
The indeterminate value is 0.1
The false value is 0.5
The unknown value is 0.1

The truth value is 0.1
The indeterminate value is 0.3
The false value is 0.2
The unknown value is 0.4
```

```
The result of NL and is
The truth value is 0.1
The indeterminate value is 0.29999998
The false value is 0.5
The unknown value is 0.1

The result of NL or is
The truth value is 0.3
The indeterminate value is 0.4
The false value is 0.2
The unknown value is 0.1

The result of NL negation is
The truth value is 0.5
The indeterminate value is 0.1
The false value is 0.3
The unknown value is 0.1

The result of NL implication is
The truth value is 0.5
The indeterminate value is 0.20000002
The false value is 0.2
The unknown value is 0.1

The result of NL joint denial is
The truth value is 0.2
The indeterminate value is 0.4
The false value is 0.3
The unknown value is 0.1

The result of NL alternative denial is
The truth value is 0.5
The indeterminate value is 0.29999998
The false value is 0.1
The unknown value is 0.1
```

The following code implements the elements of Paraconsistent Neutrosophic Logic.

```
/* Written by Charles Ashbacher June, 2002. */

public class PNLElement
{
 private float truthvalue;
 private float indeterminatevalue;
 private float falsevalue;
```

```
public PNLElement(float tvalue, float ivalue, float fvalue)
{
 float test=tvalue+ivalue+fvalue;
  if(test>=1.0)
  {
  truthvalue=tvalue;
  indeterminatevalue=ivalue;
  falsevalue=fvalue;
 }
 else
 {
  System.out.println("The inputs do not satisfy the criteria that the sum be greater than
                      or equal to 1.0");
  System.out.println("The object has been modified to have a high indeterminate value");
  truthvalue=tvalue;
  falsevalue=fvalue;
  indeterminatevalue=1.0f;
 }
}

public void updatevalues(float tvalue, float ivalue, float fvalue)
{
 float test=tvalue+ivalue+fvalue;
 if(test>=1.0)
 {
  truthvalue=tvalue;
  indeterminatevalue=ivalue;
  falsevalue=fvalue;
 }
 else
 {
  System.out.println("The inputs do not satisfy the criteria that the sum be greater than
                      or equal to 1.0");
  System.out.println("The object has been set to have a high indeterminate value");
  truthvalue=tvalue;
  falsevalue=fvalue;
  indeterminatevalue=1.0f;
 }
}

public float gettrue()
{
 return truthvalue;
}
```

```
 public float getindeterminate()
 {
  return indeterminatevalue;
 }

 public float getfalse()
 {
  return falsevalue;
 }

 public void printvalues()
 {
  System.out.println("The truth value is "+truthvalue);
  System.out.println("The indeterminate value is "+indeterminatevalue);
  System.out.println("The false value is "+falsevalue);
  System.out.println(" ");
 }

}
```

The following Java program contains functions that implement the ∧, ∨, ¬, →, ↓ and | connectives of the PNL logic.

```
/* Written by Charles Ashbacher, June, 2002. */

public class UsingPNLElement
{
  public static PNLElement PNLand(PNLElement nle1, PNLElement nle2)
  {
   float nle1true,nle1false;
   float nle2true,nle2false;
   float assigntrue,assignfalse,assignindeterminate;
   float nle1indeterminate,nle2indeterminate;

// Get the values of the components needed in the computation
   nle1true=nle1.gettrue();
   nle1false=nle1.getfalse();
   nle1indeterminate=nle1.getindeterminate();
   nle2true=nle2.gettrue();
   nle2false=nle2.getfalse();
   nle2indeterminate=nle2.getindeterminate();

// Compute the and
   assigntrue=getMin(nle1true,nle2true);
   assignfalse=getMax(nle1false,nle2false);
   assignindeterminate=getMax(nle1indeterminate,nle2indeterminate);
```

```
  PNLElement tempNL=new PNLElement(assigntrue,assignindeterminate,assignfalse);
  return tempNL;
  }

  public static PNLElement PNLor(PNLElement nle1, PNLElement nle2)
  {
  float nle1true,nle1false;
  float nle2true,nle2false;
  float nle1indeterminate,nle2indeterminate;
  float assigntrue,assignfalse,assignindeterminate;

// Get the values to be used in the computation
  nle1true=nle1.gettrue();
  nle1false=nle1.getfalse();
  nle1indeterminate=nle1.getindeterminate();
  nle2true=nle2.gettrue();
  nle2false=nle2.getfalse();
  nle2indeterminate=nle2.getindeterminate();

// Compute the or
  assigntrue=getMax(nle1true,nle2true);
  assignfalse=getMax(nle1false,nle2false);
  assignindeterminate=getMin(nle1indeterminate,nle2indeterminate);
  PNLElement tempNL=new PNLElement(assigntrue,assignindeterminate,assignfalse);
  return tempNL;
  }


  public static PNLElement PNLnegation(PNLElement nle1)
  {
  float assigntrue=nle1.getfalse();
  float assignfalse=nle1.gettrue();
  float assignindeterminate=nle1.getindeterminate();
  PNLElement tempNL=new PNLElement(assigntrue,assignindeterminate,assignfalse);
  return tempNL;
  }

  public static PNLElement PNLimplication(PNLElement nle1,PNLElement nle2)
  {
  PNLElement nle3=PNLnegation(nle1);
  PNLElement nle4=PNLor(nle3,nle2);
  return(nle4);
  }

  public static PNLElement PNLjointdenial(PNLElement nle1,PNLElement nle2)
  {
```

```
 PNLElement nle3=PNLor(nle1,nle2);
 PNLElement nle4=PNLnegation(nle3);
 return (nle4);
}

public static PNLElement PNLaltdenial(PNLElement nle1,PNLElement nle2)
{
 PNLElement nle3=PNLand(nle1,nle2);
 PNLElement nle4=PNLnegation(nle3);
 return (nle4);
}

public static float getMin(float x1,float x2)
{
 float theMin;
 if(x1>=x2)
 {
  theMin=x2;
 }
 else
 {
  theMin=x1;
 }
 return theMin;
}

public static float getMax(float x1,float x2)
{
 float theMax;
 if(x1>=x2)
 {
  theMax=x1;
 }
 else
 {
  theMax=x2;
 }
 return theMax;
}

public static void main(String args[])
{
 PNLElement nle1=new PNLElement(0.6f,0.1f,0.5f);
 PNLElement nle2=new PNLElement(0.1f,0.5f,0.7f);
 PNLElement nle3;
 nle3=PNLand(nle1,nle2);
```

```
 System.out.println("The original values are ");
 nle1.printvalues();
 nle2.printvalues();
 System.out.println("The result of NL and is ");
 nle3.printvalues();
 nle3=PNLor(nle1,nle2);
 System.out.println("The result of NL or is ");
 nle3.printvalues();
 nle3=PNLnegation(nle1);
 System.out.println("The result of NL negation is ");
 nle3.printvalues();
 nle3=PNLimplication(nle1,nle2);
 System.out.println("The result of NL implication is ");
 nle3.printvalues();
 nle3=PNLjointdenial(nle1,nle2);
 System.out.println("The result of NL joint denial is ");
 nle3.printvalues();
 nle3=PNLaltdenial(nle1,nle2);
 System.out.println("The result of NL alternative denial is ");
 nle3.printvalues();
 }
}
```

The output when this program is run is as follows.

```
The original values are
The truth value is 0.6
The indeterminate value is 0.1
The false value is 0.5

The truth value is 0.1
The indeterminate value is 0.5
The false value is 0.7

The result of NL and is
The truth value is 0.1
The indeterminate value is 0.5
The false value is 0.7

The result of NL or is
The truth value is 0.6
The indeterminate value is 0.1
The false value is 0.7

The result of NL negation is
The truth value is 0.5
The indeterminate value is 0.1
```

```
The false value is 0.6

The result of NL implication is
The truth value is 0.5
The indeterminate value is 0.1
The false value is 0.7

The result of NL joint denial is
The truth value is 0.7
The indeterminate value is 0.1
The false value is 0.6

The result of NL alternative denial is
The truth value is 0.7
The indeterminate value is 0.5
The false value is 0.1
```

Section 6
Ordering The Elements of Neutrosophic Logic

Our next step in the analysis of the elements of NL is to impose an ordering relation R on them. In considering the definition of such a relation for NL1, the truth value is considered more significant than all the others, with the indeterminate the least. With this as the foremost consideration, the following is the definition of an order relation on NL1.

Definition 4.6.1: Let (t_1, i_1, f_1) and (t_2, i_2, f_2) be elements of NL1. Then $(t_1, i_1, f_1) <_{NL1} (t_2, i_2, f_2)$ if one of the following is true:

(i) $t_1 < t_2$
(ii) $t_1 = t_2$ and $f_1 < f_2$

The expression $(t_1, i_1, f_1) \leq_{NL1} (t_2, i_2, f_2)$ is used if $t_1 = t_2$, $i_1 = i_2$ and $f_1 = f_2$ or $(t_1, i_1, f_1) <_{NL1} (t_2, i_2, f_2)$.

In other words one element in NL1 is greater than another if its truth value is larger or its false value is greater if the truth values are equal.

This definition is consistent with the idea that one element is larger than another if it is known with greater precision than another. In general, reasoning is executed by applying what is known to be true, so that is the first point of emphasis. In the case where the truth values are equal, the interest is in which is most precise, which would be the largest sum to t and f.

For NL2, the indeterminate value is considered the most significant, with the false the least significant.

Definition 4.6.2: Let (t_1, i_1, f_1) and (t_2, i_2, f_2) be elements of NL2. Then $(t_1, i_1, f_1) <_{NL2} (t_2, i_2, f_2)$ if one of the following is true:

i) $i_1 < i_2$
ii) $i_1 = i_2$ and $t_1 < t_2$

The expression $(t_1, i_1, f_1) \leq_{NL2} (t_2, i_2, f_2)$ is used if $t_1 = t_2$, $i_1 = i_2$ and $f_1 = f_2$ or $(t_1, i_1, f_1) <_{NL2} (t_2, i_2, f_2)$.

In other words one element in NL2 is greater than another if its indeterminate value is larger or its true value is greater if the indeterminate values are equal.

Definition 4.6.3: Let (t_1, i_1, f_1, u_1) and (t_2, i_2, f_2, u_2) be elements of INL1. Then $(t_1, i_1, f_1, u_1) <_{INL1} (t_2, i_2, f_2, u_2)$ if one of the following is true:

i) $t_1 < t_2$.
ii) $t_1 = t_2$ and $f_1 < f_2$.
iii) $t_1 = t_2$, $f_1 = f_2$ and $i_1 < i_2$.

The expression $(t_1, i_1, f_1, u_1) \leq_{INL1} (t_2, i_2, f_2, u_2)$ is used if $t_1 = t_2$, $i_1 = i_2$, $f_1 = f_2$ and $u_1 = u_2$ or $(t_1, i_1, f_1, u_1) <_{INL1} (t_2, i_2, f_2, u_2)$.

Definition 4.6.4: Let (t_1, i_1, f_1, u_1) and (t_2, i_2, f_2, u_2) be elements of INL2. Then $(t_1, i_1, f_1, u_1) <_{INL2} (t_2, i_2, f_2, u_2)$ if one of the following is true:

i) $t_1 < t_2$.
ii) $t_1 = t_2$ and $f_1 < f_2$.
iii) $t_1 = t_2$, $f_1 = f_2$ and $u_1 < u_2$.

The expression $(t_1, i_1, f_1, u_1) \leq_{INL2} (t_2, i_2, f_2, u_2)$ is used if $t_1 = t_2$, $i_1 = i_2$, $f_1 = f_2$ and $u_1 = u_2$ or $(t_1, i_1, f_1, u_1) <_{INL2} (t_2, i_2, f_2, u_2)$.

Definition 4.6.5: Let (t_1, i_1, f_1) and (t_2, i_2, f_2) be elements of PNL1. Then $(t_1, i_1, f_1) <_{PNL1} (t_2, i_2, f_2)$ if one of the following is true:

i) $t_1 < t_2$.
ii) $t_1 = t_2$ and $f_1 < f_2$.
iii) $t_1 = t_2$, $f_1 = f_2$ and $i_1 < i_2$.

The expression $(t_1, i_1, f_1) \leq_{PNL1} (t_2, i_2, f_2)$ is used if $t_1 = t_2$, $i_1 = i_2$ and $f_1 = f_2$ or $(t_1, i_1, f_1, u_1) < (t_2, i_2, f_2, u_2)$.

Theorem 4.6.1:
a) The order relation (<) as defined in NL1 is

(i) irreflexive,
(ii) asymmetric,
(iii) transitive,
(iv) connected.

b) The order relation (<) as defined in NL2 is

(v) irreflexive,
(vi) asymmetric,
(vii) transitive,
(viii) connected.

c) The order relation (<) as defined in INL1 is

(ix) irreflexive,
(x) asymmetric,
(xi) transitive,
(xii) connected.

d) The order relation (<) as defined in INL2 is

(xiii) irreflexive,
(xiv) asymmetric,
(xv) transitive,
(xvi) connected.

e) The order relation (<) as defined in PNL1 is

(xvii) irreflexive,
(xviii) asymmetric,
(xix) transitive,
(xx) connected.

Proof: (a)
(i) If (NL1, <) were not irreflexive, then there would have to be an element (t, i, f) such that

(t,i,f) < (t,i,f)

Clearly, it is not possible for t < t or for f < f, so this is impossible.

ii) Assume that there are elements (t_1, i_1, f_1) and (t_2, i_2, f_2) such that
$(t_1, i_1, f_1) < (t_2, i_2, f_2)$ and $(t_1, i_1, f_1) < (t_2, i_2, f_2)$.

Case 1: $t_1 \neq t_2$

By definition, this would mean that $t_1 < t_2$ and $t_1 > t_2$, which is impossible.

Case 2: $t_1 = t_2$

By definition, this would mean that $f_1 < f_2$ and $f_1 > f_2$ which is impossible.

Therefore, < on NL1 is asymmetric.

iii) Let $(t_1, i_1, f_1) < (t_2, i_2, f_2)$ and $(t_2, i_2, f_2) < (t_3, i_3, f_3)$. There are several cases to consider.

Case 1: $t_1 = t_2 = t_3$

Therefore, the inequalities imply that $f_1 < f_2$ and $f_2 < f_3$, which by the transitivity of < on real numbers, yields $f_1 < f_3$. Therefore, by definition $(t_1, i_1, f_1) < (t_3, i_3, f_3)$.

Case 2: $t_1 = t_2$ and $t_2 < t_3$

Then, $t_1 < t_3$ and $(t_1, i_1, f_1) < (t_3, i_3, f_3)$ by definition.

Case 3: $t_1 < t_2$ and $t_2 = t_3$

Then, $t_1 < t_3$ and $(t_1, i_1, f_1) < (t_3, i_3, f_3)$ by definition.

Case 4: $t_1 < t_2$ and $t_2 < t_3$

By the transitivity of < on real numbers, it follows that $t_1 < t_3$ and $(t_1, i_1, f_1) < (t_3, i_3, f_3)$ by definition.

iv) Let (t_1, i_1, f_1) and (t_2, i_2, f_2) be elements of NL1. If they are not equal, then there are two cases.

Case 1: $t_1 = t_2$ and $f_1 \neq f_2$

By the trichotomy property of real numbers, it would then follow that $f_1 > f_2$ or $f_1 < f_2$. If $f_1 > f_2$, then
$(t_1, i_1, f_1) > (t_2, i_2, f_2)$ and if $f_1 < f_2$ then $(t_1, i_1, f_1) < (t_2, i_2, f_2)$.

Case 2: $t_1 \neq t_2$

By the trichotomy property again, this would mean that $t_1 > t_2$ or $t_1 < t_2$. If $t_1 > t_2$, then
$(t_1, i_1, f_1) > (t_2, i_2, f_2)$ and if $t_1 < t_2$ then $(t_1, i_1, f_1) < (t_2, i_2, f_2)$.

The proofs of (b) through (e) are similar to that for (a) and so are omitted.

Theorem 4.6.2: If A is an element of NL1, then A $\leq_{NL1}$ (1,0,0) and A $\geq_{NL1}$ (0,1,0). This of course means that the elements of NL1 have an upper and lower bound.

Proof: Let A = (t_1, i_1, f_1). If $t_1 = 1$, then A = (1, 0, 0) and if $t_1 < 1$, then A < (1, 0, 0) by definition. If $i_1 = 1.0$, then A = (0,1,0) and if $i_1 \neq 1.0$, then one or both of t_1 or f_1 is greater than zero. In either case, this would make A > (0,1,0).

Theorem 4.6.3: If A is an element of NL2, then A $\leq_{NL2}$ (0,1,0) and A $\geq_{NL2}$ (0,0,1). This of course means that the elements of NL2 have an upper and lower bound.

Proof: Let A = (t_1, i_1, f_1). If $i_1 = 1$, then A = (0, 1, 0) and if $i_1 < 1$, then A < (0, 1, 0) by definition. If $f_1 = 1.0$, then A = (0,0,1) and if $f_1 \neq 1.0$, then one or both of i_1 or t_1 is greater than zero. In either case, this would make A > (0,1,0).

Theorem 4.6.4:
i) If A is an element of INL1 , then A $\leq_{INL1}$ (1, 0, 0, 0) and A $\geq_{INL1}$ (0, 1, 0, 0). Therefore, the elements of INL1 have an upper and a lower bound.

ii) If A is an element of INL2 , then A $\leq_{INL2}$ (1, 0, 0, 0) and A $\geq_{INL2}$ (0, 0, 0, 1). Therefore, the elements of INL2 have an upper and a lower bound.

Proof: The result is a direct consequence of the definition of $\leq_{INL1}$ and $\leq_{INL2}$.

Theorem 4.6.5: If the value of true in PNL1 is bounded above by maxt, then PNL1 has both an upper and lower bound.

Proof: Let A = (t_1, i_1, f_1) be an element of PNL1. Since $t_1 \leq$ maxt, it follows that A $\leq_{PNL1}$ (maxt, 0, 0). By definition of the elements in PNL1, $t_1 + i_1 + f_1 \geq 1.0$, so A $\leq_{PNL1}$ (0, 0, 1).

Section 7
Rules Of Inference in NL1

The Neutrosophic Logic can be used to construct a system of reasoning where premises can be used to infer or justify conclusions. To begin this process, it is necessary to define what is meant by infer and the notation used to represent it.

Definition 4.7.1: Suppose that A = (t_1, i_1, f_1) and that A $\rightarrow$ B = (t_2, i_2, f_2) in NL1. Then we can infer B with the values B = (t_3, i_3, f_3) where

$t_3 = t_2$ if $t_2 > f_1$

$t_3 = 0.0$ if $f_1 \leq t_2$

$f_3 = f_2$ if $f_2 < t_1$

$f_3 = 0.0$ if $f_2 \geq t_1$.

$i_3 = 1.0 - t_3 - f_3$

This inference rule is the modus ponens rule for NL1 (MPNL1).

The rationale for this rule is as follows. Start with the values for $A = (t_1, i_1, f_1)$ and the definition of the implication in terms of the negation and conjunction.

$A \rightarrow B = \neg A \vee B = (t_2, i_2, f_2)$.

If $B = (t_3, i_3, f_3)$, then $\max\{ f_1, t_3 \} = t_2$ and $\min\{ t_1, f_3 \} = f_2$. If $t_2 > f_1$, then it follows that $t_2 = t_3$. However, if $f_1 \leq t_2$, then we have no information about t_3, so we make no assumptions and set it equal to zero. If $f_2 < t_1$, then it follows that $f_3 = f_2$. Again, if $f_2 \geq t_1$, we have no information about the value of f_3, so we set the value to zero. The computation of i_3 is done in the usual way by taking the difference from 1.0.

Examples:
Let $A = (1, 0, 0)$ and $A \rightarrow B = (1, 0, 0)$. Then we can infer $B = (1, 0, 0)$ by applying MPNL. This is consistent with the modus ponens rule for classical logic, where if A and $A \rightarrow B$ are true, we can infer that B is also true.

If $A = (0, 0, 1)$ and $A \rightarrow B = (1, 0, 0)$, then the definition would yield $B = (1, 0, 0)$. This is also consistent with the rule of classical logic that a false hypothesis can be used to prove anything.

Definition 4.7.2: Suppose that $A = (t_1, i_1, f_1)$ and that $A \rightarrow B = (t_2, i_2, f_2)$ in NL2. Then we can infer B with the values $B = (t_3, i_3, f_3)$ where

$i_3 = i_2$ if $i_2 \geq i_1$

$i_3 = 1.0$ if $i_1 > i_1$

$t_3 = 0.0$ if $i_3 = 1.0$

$t_3 = t_2$ if $t_2 \geq t_1$ and $t_2 < 1 - i_3$

$t_3 = 0.0$ if $t_2 \geq 1 - i_3$ or $t_2 \leq 1 - i_3$

$f_3 = 1.0 - i_3 - t_3$

This inference rule is the modus ponens rule for NL2 (MPNL2).

The rationale for this rule is similar to that of MPNL1. Start with the values for $A = (t_1, i_1, f_1)$ and the definition of the implication in terms of the negation and conjunction.

$A \rightarrow B = \neg A \vee B = (t_2, i_2, f_2)$.

If $B = (t_3, i_3, f_3)$, then $\max \{ i_1, i_3 \} = i_2$. If $i_2 \geq i_1$ then it follows that i_3 must be i_2. If $i_I > i_2$, then we know nothing about the value of i_3, so we set it to 1.0.

The computation of the true value is given by

$\min \{ 1 - i_3, \max\{t_1, t_3\} \} = t_2$.

If $t_2 \geq t_1$ and $t_2 < 1 - i_3$ it follows that the value of t_2 is that of t_3, Nothing can be inferred about the value of t_3 for the other cases, so it is set to 0.0.

Let $A = (t_1, i_1, f_1, u_1)$ and $A \rightarrow B = (t_2, i_2, f_2, u_2)$ be elements of INL1. Using the alternate form of the implication, letting $B = (t_3, i_3, f_3, u_3)$ and applying the definitions of $\neg$ and $\vee$ for INL1, we have

$(f_1, i_1, t_1, u_1) \vee (t_3, i_3, f_3, u_3) = (t_2, i_2, f_2, u_2)$,

so

$\max\{ t_1, t_3 \} = t_2$

$\min\{ i_1, i_3 \} = i_2$

$\min\{ f_1, f_3 \} = f_2$

$u_2 = 1 - t_2 - i_2 - f_2$.

Which gives us a natural way to define modus ponens in INL1.

Definition 4.7.3: Given $A = (t_1, i_1, f_1, u_1)$ and $A \rightarrow B = (t_2, i_2, f_2, u_2)$ elements of INL1, we can infer $B = (t_3, i_3, f_3, u_3)$ with the following values

$t_3 = t_2$ if $t_2 \geq t_1$

$t_3 = 0.0$ if $t_2 < t_1$

$i_3 = i_2$ if $i_2 \leq i_1$

$i_3 = 0.0$ if $i_2 > i_1$

$f_3 = f_2$ if $f_2 \leq f_1$

$f_3 = 0.0$ if $f_2 > f_1$

$u_3 = 1 - t_3 - i_3 - f_3$.

This inference rule is the modus ponens rule for INL1 (MPINL1).

The rationale for this definition of modus ponens is similar to that for NL1 and NL2.

Definition 4.7.4: Given $A = (t_1, i_1, f_1, u_1)$ and $A \rightarrow B = (t_2, i_2, f_2, u_2)$ elements of INL2, we can infer $B = (t_3, i_3, f_3, u_3)$ with the following values

$t_3 = t_2$ if $t_2 \geq t_1$

$t_3 = 0.0$ if $t_2 < t_1$

$u_3 = u_2$ if $u_2 \leq u_1$

$u_3 = 0.0$ if $u_2 > u_1$

$f_3 = f_2$ if $f_2 \leq f_1$

$f_3 = 0.0$ if $f_2 > f_1$

$i_3 = 1 - t_3 - u_3 - f_3$.

This inference rule is the modus ponens rule for INL2 (MPINL2).

The differences in the definitions of modus ponens for INL1 and INL2 are due to the different roles that the unknown and indeterminate parts play. For INL1, the unknown is the value computed from all the others and hence has the lowest significance. In INL2, it is the indeterminate value that is computed from all the others.

Let $A = (t_1, i_1, f_1)$ and $A \rightarrow B = (t_2, i_2, f_2)$ be elements of PNL1. Using the equivalent form of the implication ($\neg A \vee B$) and letting $B = (t_3, i_3, f_3)$, applying the definitions of the connectives, we have the following expressions.

$\max \{ t_1, t_3 \} = t_2$

$\max \{ i_1, i_3 \} = i_2$

$\min \{ f_1, f_3 \} = f_2$.

Which provides the rationale for the definition of modus ponens in PNL1.

Definition 4.7.5: Given $A = (t_1, i_1, f_1)$ and $A \rightarrow B = (t_2, i_2, f_2)$ elements of PNL1, we can infer $B = (t_3, i_3, f_3)$ with the following values

$t_3 = t_2$ if $t_2 \geq t_1$

$t_3 = 0.0$ if $t_2 < t_1$

$f_3 = f_2$ if $f_2 \leq f_1$

$f_3 = 0.0$ if $f_2 > f_1$
$i_3 = \max \{ 1.0 - t_3 - f_3, \max\{i_1, i_2 \} \}$.

In this case, since the sum of the values must be at least 1.0, it follows that the potential for the true and false portions to be zero requires that the definition of the indeterminate value yield value that is at least 1.0.

Section 8

Formal Theories In Neutrosophic Logic

Definition 4.8.1: The following is the definition of the formal theory LNL1 in the Neutrosophic Logic.

a) The set of symbols { ¬, ∨, ∧, →, (,), T, F, A, B, . . . }, where the symbols A, B, . . . are abbreviations for triplets of the form (t, i, f) where t + i + f = 1.0 and the constants T, I and F which are abbreviations for (1, 0, 0), (0, 1, 0) and (0, 0, 1) respectively.
b) 1) The constants T, I and F are well-formed.
2) The symbols A, B, . . . are well-formed.
3) If A is well-formed, then so is ¬A and (A).
4) If A and B are well-formed, then so are A ∨ B, A ∧ B and A → B.
5) Only expressions that can be formed using rules 1 – 4 are well-formed.
c) If A and B are well-formed, then the following are the axioms of LNL1.

1) A → (A ∨ B) has the value (1, 0, 0). LNL1A1
2) (A ∧ B) → A has the value (1, 0, 0). LNL1A2
3) (A ∧ B) → B has the value (1, 0, 0). LNL1A3

d) The only rule of inference is MPNL1.

Definition 4.8.2: The formal theory LNL2 can be defined by taking properties (a) through (c) of LNL1 with suitable renaming and the following alternate definition of (d).

d) The only rule of inference is MPNL2.

Definition 4.8.3: The following is the definition of the formal theory INL1 in the Neutrosophic Logic.

a) The set of symbols { ¬, ∨, ∧, →, (,), T, F, A, B, . . . }, where the symbols A, B, . . . are abbreviations for 4-tuples of the form (t, i, f, u) where t + i + f+ u = 1.0 and the constants T, I, F and U which are abbreviations for (1, 0, 0,0), (0, 1, 0,0), (0, 0, 1,0) and (0, 0, 0, 0, 1) respectively.

b) 1) The constants T, I, F and U are well-formed.
2) The symbols A, B, . . . are well-formed.
3) If A is well-formed, then so is ¬A and (A).
4) If A and B are well-formed, then so are A ∨ B, A ∧ B and A → B.
5) Only expressions that can be formed using rules 1 – 4 are well-formed.

c) If A and B are well-formed, then the following are the axioms of INL1.

1) A → (A ∨ B) has the value (1, 0, 0), 0). INL1A1
2) (A ∧ B) → A has the value (1, 0, 0, 0). INL1A2
3) (A ∧ B) → B has the value (1, 0, 0, 0). INL1A3

d) The only rule of inference is MPINL1.

Definition 4.8.4: The formal theory INL2 can be defined by taking properties (a) through (c) of ILNL1 with suitable renaming and the following alternate definition of (d).

d) The only rule of inference is MPINL2.

Definition 4.8.5: The following is the definition of the formal theory PNL1 in the Neutrosophic Logic.

a) The set of symbols { ¬, ∨, ∧, →, (,), A, B, . . . }, where the symbols A, B, . . . are abbreviations for triplets of the form (t, i, f) where t + i + f ≥ 1.0.

b) 1) The symbols A, B, . . . are well-formed.
2) If A is well-formed, then so is ¬A and (A).
3) If A and B are well-formed, then so are A ∨ B, A ∧ B and A → B.
4) Only expressions that can be formed using rules 1 – 4 are well-formed.

c) If A and B are well-formed, then the following are the axioms of PNL1.

1) A → (A ∨ B) has a value where the true component is greater than or equal to 1.0. PNL1A1
2) (A ∧ B) → A has a value where the true component is greater than or equal to 1.0. PNL1A2
3) (A ∧ B) → B has a value where the true component is greater than or equal to 1.0. PNL1A3

d) The only rule of inference is MPPNL1.

Theorem 4.8.1: If A → B has the value (1, 0, 0) in LNL1 (LNL2) and A ≠ F, then we can infer B = (1, 0, 0) in LNL1 (LNL2).

Proof: Let A = (t, i, f) ≠ F be an arbitrary element of LNL1. Therefore, f must be less than 1, so by MPNL1 the first element of the triplet of B is the first element of A → B, which is 1. By definition, the other two values of the triplet must be zero and B = (1, 0, 0). Since the definition of MPNL2 is the same as that of MLNL1 for the true and false component, the theorem also holds for LNL2.

Corollary: For A and B elements of LNL1(LNL2).
i) If A ≠ F, then we can infer (A ∨ B) = T.
ii) If (A ∧ B) ≠ F, then we can infer A = T.
iii) If (A ∧ B) ≠ F, then we can infer B = T.

Proof:
i) Apply theorem 4.8.1 on LNL1A1 (LNL2A1).
ii) Apply theorem 4.8.1 on LNL1A2 (LNL2A2).
iii) Apply theorem 4.8.1 on LNL1A3 (LNL2A3).

Theorem 4.8.2: If A → B has the value (1, 0, 0, 0) in INL1 (INL2) and A ≠ F, then we can infer B = (1, 0, 0, 0) in INL1 (INL2).

Proof: Let A = (t, i, f, u) ≠ F be an arbitrary element of INL1. Therefore, f must be less than 1, so by MPINL1 the first element of the 4-tuple of B is the first element of A → B, which is 1. By definition, the other three values of the 4-tuple must be zero and B = (1, 0, 0, 0). Since the definition of MPINL2 is the same as that of MPINL1 for the true and false component, the theorem also holds for INL2

Theorem 4.8.3: If A → B has a true value that is greater than or equal to 1.0 in PNL1 and the false value of A is less than 1.0, then we can infer that B has a true value greater than or equal to 1.0.

Proof: Let A = (t, i, f) where f < 1.0, then the true value of ¬A must be less than 1.0. Therefore, since the true value of ¬A ∨ B is greater than or equal to 1.0, it follows from MPPNL1 that the true value of B must be greater than or equal to 1.0.

Theorems 4.8.1, 4.8.2 and 4.8.3 point out some of the difficulties that exist when defining axioms for Neutrosophic Logic. By assigning the axioms values of (1, 0, 0), all we need are nonzero antecedents to prove that the consequence is T. This is a stronger result than we may want or expect.

Example:
Construct the formal theory LNL12 by using the definition of LNL1, except use the following axiom list instead.

(A → (B → A)) has the value T. LNL12A1

((A → (B → C)) → ((A → B) → (A → C))) has the value T. LNL12A2
((¬B → ¬A) → ((¬B → A) → B)) has the value T. LNL12A3.

Theorem 4.8.4: If A, B and C are elements of LNL12:
i) If A ≠ F, then (B → A) has the value T.
ii) If ((A → (B → C)) ≠ F, then ((A → B) → (A → C)) has the value T.
iii) If (¬B → ¬A) ≠ F, then ((¬B → A) → B) has the value T.

Proof:
Identical to that of theorem 4.8.1.

Theorem 4.8.5: If A, B, and C are elements of LNL2:
i) If A≠ T, then ((A → B) → (A → C)) has the value T.
ii) If B ≠ T or C ≠ F, then ((A → B) → (A → C)) has the value T.
iii) If B ≠ F or A ≠ T, then ((¬B → A) → B) has the value T.

Proof:
i) Since A ≠ T, we can apply the definition of the ∨ and ¬ connectives to conclude that ¬A ∨ (B → C) is not F. Since this is an abbreviation for A → (B ∧ C), this expression is also not F. We then apply theorem 4.8.4 with LNL2A2 to infer that
((A → B) → (A → C)) has the value T.

ii) If B ≠ T or C ≠ F, then ¬A ∨ (¬B ∨ C) is not false. This expression is an abbreviation for A → (B → C), so it also is not false. We can then apply theorem 58 with LNL2A2 to conclude that ((A → B) → (A → C)) has the value T.

iii) If B ≠ F, then ¬B ≠ T and if A ≠ F, then ¬A ≠ T. Therefore, B ∨ ¬A ≠ F and by abbreviation ¬B → ¬A ≠ F. We can then apply theorem 58 with LNL2A3 to conclude that ((¬B → A) → B) has the value T.

The alternative set of axioms LNL12A1, LNL12A2 and LNL12A3 can also be used to create additional forma theories with INL1, INL2 and PNL1. Since all are based on the representation of A → B as ¬A ∨ B and involve the value of T, theorem 4.8.5 holds for INL1, INL2 and PNL1 where the axiom list is replaced by

(A → (B → A)) has the value T.
((A → (B → C)) → ((A → B) → (A → C))) has the value T.
((¬B → ¬A) → ((¬B → A) → B)) has the value T.

Section 9

Reasoning in Neutrosophic Logic

Definition 4.9.1: If A and B are well-formed expressions in LNL1, LNL2, INL1, INL2 or PNL1 then **B is said to be a consequence of A**, noted by A $\vdash$ B, if the value of expression B can be inferred to be greater than or equal to that of expression A. An equivalent expression would be that B can be inferred from A. The expression

$$\vdash A$$

holds only when A = T. In the case of A $\vdash$ B, A is the **hypothesis** and B the **conclusion**. The sequence of steps that start with A and lead to B are known as a **proof**.

Definition: If A and B are well-formed expressions in PNL1, then **B is said to be a consequence of A**, noted by A $\vdash$ B, if the value of expression B can be inferred to be greater than or equal to that of expression A. An equivalent expression would be that B can be inferred from A. The expression

$$\vdash A$$

holds only when the true value of A is greater than or equal to 1.0.

Theorem 4.9.1: If A is an element of LNL1, LNL2, INL1, INL2 or PNL1, then

(i) $A \vdash A$
(ii) $A \vee A \vdash A$
(iii) $A \wedge A \vdash A$
(iv) $A \vdash A \vee A$
(v) $A \vdash A \wedge A$
(vi) $A \vdash \neg\neg A$
(vii) $\neg\neg A \vdash A$

Proof: (For LNL1 only).
(i) Since $A \leq_{NL1} A$, the definition is satisfied.
(ii) It has been proven that $A \vee A = A$, so the problem simplifies to case (i).
(iii) If has been proven that $A \wedge A = A$, so the problem simplifies to case (i).
(iv) Similar to case (ii).
(v) Similar to case (iii).
(vi) It has been proven that $\neg\neg A = A$, so the problem simplifies to case (i).
(vii) Similar to case (vi).

Theorem 4.9.2: If A, B and C are elements of LNL1, LNL2, INL1, INL2 or PNL1 then the $\vdash$ operator is

i) $A \vdash A$. (Reflexive.)
ii) If $A \vdash B$ and $B \vdash A$ then B = A. (Antisymmetric.)
iii) If $A \vdash B$ and $B \vdash C$ then $A \vdash C$. (Transitive.)

Proof: (For LNL1 only).

i) This is the result of theorem 4.9.1 part (i).

ii) If A $\vdash$ B then A $\leq_{NL1}$ B and if B $\vdash$ A then B $\leq_{NL1}$ A. This can hold if and only if A = B.

iii) If A $\vdash$ B then the expression A can be used to infer that the value of B is greater than or equal to A. If B $\vdash$ C then the expression B can be used to infer that the value of C is greater than or equal to B. First perform the steps that will start at A and lead to B. Follow this by performing the steps that start at B and lead to C. The combination will then be a sequence of steps that will start at A and lead to C. Therefore, A can be used to infer C and A $\vdash$ C.

Theorem 4.9.3: If A is any wff of LNL1 or LNL2.

(i) (0, 1, 0) $\vdash$ A.
(ii) A $\vdash$ (1, 0, 0).
(iii) (1, 0, 0) $\vdash$ A if and only if A = (1, 0, 0).
(iv) If A = (t, i, f) then (0, 0, 1) $\vdash$ A if and only if t > 0.0 or A = (0, 0, 1).

Proof: Proof for LNL1 only, the proof for LNL2 is similar.

(i) Since (0, 1, 0) $\leq_{NL1}$ A for all A in LNL1, the result is immediate.
(ii) Since (1, 0, 0) $\geq$ A for all A in LNL1, the result is immediate.
(iii) (If) Since (1, 0, 0) $\geq$ A with equality if and only if A = (1, 0, 0), the result is immediate.
(Only if) If A = (1, 0, 0) the result follows from a previous theorem.
(iv) (If) Assume that (0, 0, 1) $\vdash$ A and suppose that t =0.0 and A $\neq$ (0, 0, 1). Then i > 0.0 and f < 1.0. By definition, A < (0, 0, 1) and cannot be inferred from (0, 0, 1). Therefore, t must be greater than zero or A = (0, 0, 1).
(Only if) If t > 0.0 or A = (0, 0, 1), then A $\geq$ (0, 0, 1) and (0, 0, 1) $\vdash$ A.

Theorem 4.9.4: If A is any wff of INL1.

(i) (0, 0, 0, 1) $\vdash$ A.
(ii) A $\vdash$ (1, 0, 0, 0).
(iii) (1, 0, 0, 0) $\vdash$ A if and only if A = (1, 0, 0, 0).
(iv) If A = (t, i, f, u) then (0, 0, 1, 0) $\vdash$ A if and only if t > 0.0 or A = (0, 0, 1, 0).

Proof:
i) Since (0, 0, 0, 1) $\leq_{ILNL1}$ A for all A in INL1, the result is immediate.
ii) Since (1, 0, 0, 0) $\geq_{ILNL1}$ A for all A in INL1, the result is immediate.
iii) (If) Since (1, 0, 0, 0) $\geq_{ILNL1}$ A with equality if and only if A = (1, 0, 0, 0), the result is immediate.
(Only if) If A = (1, 0, 0, 0) the result follows from a previous theorem.
iv) (If) Assume that (0, 0, 1, 0) $\vdash$ A and suppose that t =0.0 and A $\neq$ (0, 0, 1, 0). Then i > 0.0 and f < 1.0. By definition, A $<_{ILNL1}$ (0, 0, 1, 0) and cannot be inferred from (0, 0, 1, 0). Therefore, t must be greater than zero or A = (0, 0, 1, 0).
(Only if) If t > 0.0 or A = (0, 0, 1, 0), then A $\geq_{ILNL1}$ (0, 0, 1, 0) and (0, 0, 1, 0) $\vdash$ A.

Theorem 4.9.5: If A is any wff of INL2.

(i) $(0, 1, 0, 0) \vdash A$.
(ii) $A \vdash (1, 0, 0, 0)$.
(iii) $(1, 0, 0, 0) \vdash A$ if and only if $A = (1, 0, 0, 0)$.
(iv) If $A = (t, i, f, u)$ then $(0, 0, 1, 0) \vdash A$ if and only if $t > 0.0$ or $A = (0, 0, 1, 0)$.

Proof:
i) Since $(0, 1, 0, 0) \leq_{ILNL2} A$ for all A in INL2, the result is immediate.
The proofs of (ii) through (iv) are similar to those of theorem 4.9.4.

Theorem 4.9.6: If A is any wff of PNL1,

(i) $(0, 0, 0) \vdash A$.
(ii) If there is a maximum value t_{max} for the true value in PNL1, then $A \vdash (t_{max}, 0, 0, 0)$.

Proof:
i) Since $(0, 0, 0) \leq_{PNL1} A$ for all A in PNL1, the result is immediate.
ii) If t_{max} is the largest possible value for the true component, then $(t_{max}, 0, 0, 0) \geq_{PNL1} A$ for all A in PNL1.

The strategy used in proving the various sections of the previous theorem can be generalized, which is the point of the next theorem.

Theorem 4.9.7: Suppose that A and B are well-formed in LNL1. If $A =_{NL1} B$, then $A \vdash B$ and $B \vdash A$. The same result also holds for LNL2, INL1, INL2 and PNL1.

Proof: Assume that $A =_{NL1} B$. By definition, this means that the values of the expressions are the same. Since the value of B is greater than or equal to A, $A \vdash B$ by definition. Since the equality is reflexive, the second inference also holds. The proofs for LNL2, INL1, INL2 and PNL1 are similar and are omitted.

Corollary: If A, B and C are well-formed in LNL1, then the following are all valid inferences:

(i) $A \vee B \vdash B \vee A$.
(ii) $A \wedge B \vdash B \wedge A$.
(iii) $(A \vee B) \vee C \vdash A \vee (B \vee C)$.
(iv) $(A \wedge B) \wedge C \vdash A \wedge (B \wedge C)$.
(v) $(A \vee B) \wedge A \vdash A$,
(vi) $(A \wedge B) \vee A \vdash A$.
(vii) $(1, 0, 0) \wedge A \vdash A$.
(viii) $(0, 0, 1) \vee A \vdash A$.
(ix) $(0, 0, 1) \wedge A \vdash (0, 0, 1)$.

(x) $(1, 0, 0) \vee A \vdash (1, 0, 0)$.
(xi) $A \wedge (B \vee C) \vdash (A \wedge B) \vee (A \wedge C)$.
(xii) $(A \wedge B) \vee (A \wedge C) \vdash A \wedge (B \vee C)$.
(xiii) $A \vee (B \wedge C) \vdash (A \vee B) \wedge (A \vee C)$.
(xiv) $(A \vee B) \wedge (A \vee C) \vdash A \vee (B \wedge C)$.
(xv) $\neg (A \vee B) \vdash \neg A \wedge \neg B$.
(xvi) $\neg A \wedge \neg B \vdash \neg (A \vee B)$.
(xvii) $\neg (A \wedge B) \vdash \neg A \vee \neg B$.
(xviii) $\neg A \vee \neg B \vdash \neg (A \wedge B)$.

Similar formulas, although not all, also hold in LNL2, INL1, INL2 and PNL1.

Proof:
All of these inferences are consequences of theorems already proven concerning the algebraic properties of LNL1.

Theorem 4.9.8: If A and B are well-formed in LNL1
i) If $A <_{LNL1} B$, then $A \vdash (A \vee B)$.
ii) If $A >_{LNL1} B$, then $A \vdash (A \vee B)$ if and only if $(A \vee B) = A$.
iii) If $A <_{LNL1} B$, then $A \vdash (A \wedge B)$.
iv) If $A >_{LNL1} B$, then $A \vdash (A \wedge B)$ if and only if $(A \wedge B) = A$.

Proof:
Let $A = (t_1, i_1, f_1)$, $B = (t_2, i_2, f_2)$, $A \vee B = (t_3, i_3, f_3)$ and $A \wedge B = (t_4, i_4, f_4)$.

i) Case 1: $t_2 > t_1$.
Then $t_3 = t_2 > t_1$ and $(A \vee B) > A$.

Case 2: $t_1 = t_2$ and $f_1 < f_2$.
Then, $t_3 = t_1$ and $f_3 = f_1$, which means that $(A \vee B) = A$.

ii)
(If) Case 1: $t_1 > t_2$
Then $t_3 = t_1$ and $(A \vee B) \geq A$ if and only if $f_3 \geq f_1$. Since $f_3 = \min\{ f_1, f_2 \}$, this can occur only if $f_1 = f_3$. This would make $(A \vee B) = A$.

Case 2: $t_1 = t_2$
This would also mean that $t_3 = t_1$ and it reduces to case 1.

(Only if) If $(A \vee B) = A$, then it reduces to $A \vdash A$, which was proven in theorem 4.9.2.
iii) Case 1: $t_2 > t_1$
Then $t_3 = t_1$ and $f_3 = \max\{ f_1, f_2 \} \geq f_1$. Therefore, by definition, $(A \wedge B) > A$.

Case 2: $t_2 = t_1$
Then $t_3 = t_1$ and it reduces to case 1.

iv) (If)Case 1: $t_1 > t_2$
Then $t_3 = \min\{ t_1, t_2 \} = t_2 < t_1$ and it follows that $(A \wedge B) < A$.

Case 2: $t_1 = t_2$ and $f_1 > f_2$.
Then $f_3 = \max\{ f_1, f_2 \} = f_1$ and $A = (A \wedge B)$.

(Only if) If $(A \wedge B) = A$, then it reduces to $A \vdash A$, which was proven in theorem 4.9.2.

We can think of theorem 4.9.8 as one that allows us to "add stuff", in that it allows us to infer an expression that is larger than the original. These theorems can in fact be generalized to an arbitrary number of additions.

Theorem 4.9.10: If $A <_{NL1} B_i$ for all $\{ B_1, B_2, \ldots, B_k \}$, then

i) $A \vdash (A \vee B_1 \vee B_2 \vee \ldots \vee B_k)$.
ii) $A \vdash (A \wedge B_1 \wedge B_2 \wedge \ldots \wedge B_k)$.

Proof: The proof will rely on previous results expressed in corollaries.

Let $A = (t_1, i_1, f_I)$ and $(A \vee B_1 \vee B_2 \vee \ldots \vee B_k) = (t_2, i_2, f_2)$.
i) By corollary 1, t_2 is the largest of all the first elements of the triplets in $(A \vee B_1 \vee B_2 \vee \ldots \vee B_k)$. There are two cases to consider:

Case 1: t_1 is not the largest of the first elements of the triplets.

Then $t_2 > t_1$ and $A <_{Nl1} (A \vee B_1 \vee B_2 \vee \ldots \vee B_k)$, which implies
$A \vdash (A \vee B_1 \vee B_2 \vee \ldots \vee B_k)$.

Case 2: t_1 is equal to the largest of the first elements of the triplets.

Then $f_1 \leq f_2$ and $A <_{NL1} (A \vee B_1 \vee B_2 \vee \ldots \vee B_k)$, which implies
$A \vdash (A \vee B_1 \vee B_2 \vee \ldots \vee B_k)$.

ii) By hypotheses, since $A < B_I$ for all i, t_1 is the smallest of the first elements in $(A \wedge B_1 \wedge B_2 \wedge \ldots \wedge B_k)$ and it follows that $t_1 = t_2$. Since f_1 must be less than or equal to all of the third entries in the triplets in $(A \wedge B_1 \wedge B_2 \wedge \ldots \wedge B_k)$, $f_2 \geq f_1$. Therefore, $(A \wedge B_1 \wedge B_2 \wedge \ldots \wedge B_k) \geq A$ and $A \vdash (A \wedge B_1 \wedge B_2 \wedge \ldots \wedge B_k)$.

Theorem 4.9.11: If A and B are well-formed in LNL2
i) If $A <_{LNL2} B$, then $A \vdash (A \vee B)$.
ii) If $A >_{LNL2} B$, then $A \vdash (A \vee B)$ if and only if $(A \vee B) = A$.
iii) If $A <_{LNL2} B$, then $A \vdash (A \wedge B)$.
iv) If $A >_{LNL2} B$, then $A \vdash (A \wedge B)$ if and only if $(A \wedge B) = A$.

Proof: The proof is similar to that of theorem 4.9.8 and is also based on the definition of inequality in LNL2.

Results similar to those of theorems 4.9.8 and 4.9.9 also hold for INL1, INL2 and PNL1. The proofs are all based on the definitions of ordering of the elements in the formal theory.

Definition 4.9.2: Let $x_1, x_2, \ldots, x_k, y$ be elements of NL1. Then, y can be inferred from the set of expressions $H = \{ x_1, x_2, \ldots, x_k \}$ written $x_1, x_2, \ldots, x_k \vdash y$, if given the expressions in H, y has a value greater than or equal to $\min\{ x_1, x_2, \ldots, x_k \}$. The elements in H are known as the **hypotheses**. In this case, the inequality operator used to compute the minimum is $\leq_{NL1}$.

Similar definitions can be written with NL1 replaced by NL2, INL1, INL2 or PNL1.

Theorem 4.9.12: Let $\{ x_1, x_2, x_3, \ldots, x_k \}$ be any set of expressions in NL1 (NL2) and for the expressions in the set $x_i = (t_i, i_I, f_I)$, $1 \leq i \leq k$.

i) $x_1, x_2, x_3, \ldots, x_k \vdash x_i$, for all i, $1 \leq i \leq k$.
ii) $x_1, x_2, \ldots x_k, (0, 1, 0) \vdash y$, for any expression y.
iii) $x_1, x_2, \ldots x_k \vdash (0, 1, 0)$.
iv) $x_1, x_2, \ldots x_k \vdash (x_1 \wedge x_2 \wedge \ldots \wedge x_k)$.
v) $x_1, x_2, \ldots x_k \vdash (x_1 \vee x_2 \vee \ldots \vee x_k)$.

Proof:
i) Since $x_I \geq \min\{ x_1, x_2, \ldots, x_k \}$, the result is immediate.
ii) Since $y \geq (0, 1, 0)$ for all y in NL1 (NL2), the result is immediate.
iii) Since $x_I \geq (0, 1, 0)$ for all $1 \leq i \leq k$, the result is immediate.
iv) By definition, $(x_1 \wedge x_2 \wedge \ldots \wedge x_k) = (t, i, f)$, where $t = \min\{ t_1, t_2, \ldots, t_k \}$ and $f = \max\{ f_1, f_2, \ldots, f_k \}$. Let $x_i = \min\{ x_1, x_2, \ldots x_k \}$, where $x_I = (t_i, i_i, f_i)$. By the definition of the ordering of the elements $t = t_I$ and $f \geq f_I$, which yields $(t, i, f) \geq (t_i, i_i, f_i)$.
v) By definition, $(x_1 \vee x_2 \vee \ldots \vee x_k) = (t, i, f)$, where $t = \max(t_1, t_2, \ldots, t_k \}$ and $f = \min\{ f_1, f_2, \ldots, f_k \}$. There are two cases to consider.

Case 1: The values of $\{ t_1, t_2, \ldots, t_k \}$ are not all the same. Then there is a maximum t_{max} and a minimum t_{min}, where $t_{max} > t_{min}$. Clearly, $t = t_{max}$ and the truth value of $\min\{ x_1, x_2, \ldots, x_k \}$ must be t_{min}. Therefore, the value of $(x_1 \vee x_2 \vee \ldots \vee x_k)$ must be larger than the minimum value of the hypotheses.

Case 2: The values of $\{ t_1, t_2, \ldots, t_k \}$ are all the same. Then the hypothesis with the smallest value will be the expression with the smallest value for the false component. The expression $(x_1 \vee x_2 \vee \ldots \vee x_k)$ will then have a truth value matching that common to all the hypotheses and a false value equal to that of the hypotheses having the smallest false component. This means that $\min\{ x_1, x_2, \ldots x_k \} = (x_1 \vee x_2 \vee \ldots \vee x_k)$, which satisfies the definition of inference.

Theorem 4.9.13: Let $\{ x_1, x_2, x_3, \ldots, x_k \}$ be any set of expressions in INL1 and for the expressions in the set $x_i = (t_i, i_I, f_I , u_I)$, $1 \leq i \leq k$.

i) $x_1, x_2, x_3, \ldots, x_k \vdash x_i$, for all i, $1 \leq i \leq k$.
ii) $x_1, x_2, \ldots x_k, (0, 0, 0, 1) \vdash y$, for any expression y.
iii) $x_1, x_2, \ldots x_k \vdash (0, 0, 0, 1)$.
iv) $x_1, x_2, \ldots x_k \vdash (x_1 \wedge x_2 \wedge \ldots \wedge x_k)$.
v) $x_1, x_2, \ldots x_k \vdash (x_1 \vee x_2 \vee \ldots \vee x_k)$.

Proof: Similar to that of theorem 4.9.12.

Theorem 4.9.14: Let $\{ x_1, x_2, x_3, \ldots, x_k \}$ be any set of expressions in INL2 and for the expressions in the set $x_i = (t_i, i_I, f_I , u_I)$, $1 \leq i \leq k$.

i) $x_1, x_2, x_3, \ldots, x_k \vdash x_i$, for all i, $1 \leq i \leq k$.
ii) $x_1, x_2, \ldots x_k, (0, 1, 0, 0) \vdash y$, for any expression y.
iii) $x_1, x_2, \ldots x_k \vdash (0, 1, 0, 0)$.
iv) $x_1, x_2, \ldots x_k \vdash (x_1 \wedge x_2 \wedge \ldots \wedge x_k)$.
v) $x_1, x_2, \ldots x_k \vdash (x_1 \vee x_2 \vee \ldots \vee x_k)$.

Proof: Similar to that of theorem 4.9.12.

Theorem 4.9.15: Let $\{ x_1, x_2, x_3, \ldots, x_k \}$ be any set of expressions in PNL1 and for the expressions in the set $x_i = (t_i, i_I, f_I , u_I)$, $1 \leq i \leq k$.

i) $x_1, x_2, x_3, \ldots, x_k \vdash x_i$, for all i, $1 \leq i \leq k$.
ii) $x_1, x_2, \ldots x_k, (0, 0, 0) \vdash y$, for any expression y.
iii) If there is a maximum value (t_{max}) for the truth value of an expression in PNL1, then $x_1, x_2, \ldots x_k \vdash (t_{max} , 0, 0)$.
iv) $x_1, x_2, \ldots x_k \vdash (x_1 \wedge x_2 \wedge \ldots \wedge x_k)$.
v) $x_1, x_2, \ldots x_k \vdash (x_1 \vee x_2 \vee \ldots \vee x_k)$.

Proof: Similar to that of theorem 4.9.12.

Theorem 4.9.16: Suppose that $\{ x_1, x_2, x_3, \ldots, x_k, y, z \}$ are expressions in NL1 and further suppose that $x_1, x_2, x_3, \ldots, x_k \vdash y$ and $y = z$. Then $x_1, x_2, x_3, \ldots, x_k \vdash z$. Similar results are also true in NL2, INL1, INL2 and PNL1.

Proof: Since $\min\{ x_1, x_2, x_3, \ldots, x_k \} \leq y = z$, it follows that $\min\{ x_1, x_2, x_3, \ldots, x_k \} \leq z$ and $x_1, x_2, x_3, \ldots, x_k \vdash z$ by definition. The proof in the other theories is identical as it is based only on the ability to order the elements.

Theorem 4.9.17: Let $\{ x_1, x_2, x_3, \ldots, x_k,, y_1, y_2, \ldots, y_n, z \}$ be expressions in NL1. Assume that for all i, $1 \leq i \leq n$, $x_1, x_2, x_3, \ldots, x_k, \vdash y_i$ and that $y_1, y_2, \ldots, y_n \vdash z$. Then $x_1, x_2, x_3, \ldots, x_k, \vdash z$. Similar results are also true in NL2, INL1, INL2 and PNL1.

Proof: From $x_1, x_2, x_3, \ldots, x_k, \vdash y_i$, it follows that $\min\{ x_1, x_2, x_3, \ldots, x_k \} \leq y_I$ for all $1 \leq i \leq n$ and from $y_1, y_2, \ldots, y_n \vdash z$, $\min\{ y_1, y_2, \ldots, y_n \} \leq z$. Therefore, since $\leq$ is transitive on LNL1, it follows that $x_1, x_2, x_3, \ldots, x_k, \vdash z$.

References

1. D. Bochvar, "On Three-Valued Logical Calculus and Its Application to the Analysis of Contradiction", **Matematiceskij Sbornik**, (1939), 353-369.
2. E. V. Huntington, "Sets of independent postulates for the algebra of logic", **Trans. Amer. Math. Soc**. **5**(1904), 288-300.
3. J. Lukasiewicz, "O logice trójwartościowej" (On three-valued logic), **Ruch Filozoficzny 5**, (1920), 169-171.
4. E. Mendelson, **Introduction to Mathematical Logic, Second Edition**, D. Van Nostrand, 1979.
5. F. Smarandache, A Unifying Field in Logics: Neutrosophic Logic. Neutrosophy, Neutrosophic Set, Neutrosophic Probability and Statistics, American Research Press, 1999: http://www.gallup.unm.edu/~smarandache/eBook-Neutrosophics2.pdf
6. F. Smarandache, **Neutrosophy. Neutrosophic Probability, Set and Logic**, American Research Press, 1998.
7. F. Smarandache, editor, Proceedings of the First International Conference on Neutrosophy, Neutrosophic Logic, Set, Probability and Statistics, University of New Mexico, 1-3 December 2001: http://www.gallup.unm.edu/~smarandache/NeutrosophicProceedings.pdf .
8. A. Tarski, **Introduction to Logic**, Oxford University Press, 1941.
9. L. A. Zadeh, "Fuzzy Sets", Information and Control, (1965), 338-353.

Index

A

Absolute contradiction, 40, 93-94
Absolute tautology, 40, 93-94
Absorption, 14, 31, 38, 73, 75
Alternative denial, 11, 39, 81, 97-98
Allen, J., 52
And operation, 7
And-elimination, 17, 35, 47
And-introduction, 17, 35, 47
Antisymmetric property, 131
Associative property, 14, 30, 38, 59, 62, 64, 67, 72, 73
Asymmetric property, 120-122
Antecedent, 11, 13, 18, 129
Aristotle, 57

B

Bhattacharya, S., 52
Biconditional, 11, 25
Binary form, 22
Bitwise operations, 22
Bochvar, D., 138
Bochvar three-valued logic, 27-29, 59
Boolean algebra, 89-91
Boolean function, 7-10
Bound variable, 20
Buller, A., 52

C

C++, 22
Classes in computer programming, 40
Classical logic, 7-22
Closure property, 62, 89
Commutative property, 30, 38, 48, 63, 64, 76, 98
Complete set of connectives, 10-12
Conclusion, 130
Conditional evaluation, 28
Conditional operation, 10
Conjunction, 7, 17, 22, 26-28, 32, 60, 61, 67, 68, 70, 86, 87, 123, 124
Connected, 120, 121
Consequence, 11
Contingency, 13
Contingent truth, 48-49

Contradiction, 13, 36, 40, 86, 93, 94

D

DeMorgan's rule 14, 32, 39, 91, 92
Dezert, J., 52
Dinulescu-Campina, Gh. C., 52
Disjunction 7, 18, 22, 26-28, 32, 60, 61, 67 – 72, 88
Disjunctive normal form, 10
Distributive property, 14, 31, 38, 79, 82, 84, 86, 89, 90, 91
Dominant element, 86, 87
Domination law, 14
Dual, 86

E

Equivalence relation, 89
Exclusive or, 11, 39, 46, 95,
Existential quantifier, 20, 26, 28, 67 - 71

F

Formal reasoning, 18-19, 35, 47
Formal theory, 16, 33, 127-129, 135
Free variable, 20
Fuzzy logic, 37-52, 54, 55, 57, 59, 95

G

Gershenson, C., 52
Group, 88

H

Highly significant, 40
Highly significant contradiction, 40
Highly significant tautology, 40
Homomorphism, 56
Huntington, E. V., 138
Hypotheses, 94, 135, 136,
Hypothesis, 54, 94, 124, 130, 136

I

Idempotent law, 79
Identity element, 77, 78
Identity law, 14
Implication, 10-11, 18, 25, 39, 42, 46, 96, 98, 124, 125, 126
Increased consequence, 47
Indeterminate, 26, 29, 52, 53, 59, 60, 61, 62, 68, 85, 119, 120, 127
INL1 (Intuitionistic Neutrosophic Logic 1), 55-137
INL2 (Intuitionistic Neutrosophic Logic 2), 55-137

Interpretation, 48
Intuitionistic logic, 52, 55
Inverse, 88, 89, 91
Irreflexive property, 120, 121
Isomorphism, 53

J
Java, 22
Joint denial, 11

K
Kleene, S., 25 - 27
Khoshnevisan, M, 52

L
Law of the excluded middle, 13, 23
Laws of complementarity, 89
Le, Charles T., 52
Liu, F., 52
LNL1, 127-137
LNL2, 127-137
Logical equivalence, 14-15, 26, 43, 95
Lucas, C., 52
Lukasiewicz, Jan, 23, 138
Lukasiewicz three-valued logic, 23-25, 30, 32, 33-37, 56, 57, 59

M
Mendelson, E., 138
Modal frame, 49
Modal logic, 48
Modus ponens, 16-17, 32-35, 47, 123, 124, 125, 126
Modus tollens, 18, 35
Monoid, 77, 78, 88

N
Nand operation, 11
Necessary truth, 49
Negation operation, 7
Neutrosophic logic, 52-138
Neutrosophic logic where indeterminancy is an error range, 58
Neutrosophic probability, 54
Neutrosophic set, 54
Neutrosophic statistics, 54
Neutrosophy, 3, 52, 53, 138
NL1 (Neutrosophic Logic 1), 59-137
NL2 (Neutrosophic Logic 2), 60-137

Nonzero valid, 46
Nor operation, 11
Not operation, 7

O

Object-oriented programming, 40
One-half tautology, 36
Or-introduction, 18, 35, 47
Or operation, 7
Ordering relation, 119

P

Paraconsistent logic, 52, 55, 113
Paradox, 27, 29, 54
PNL1 (Paraconsistent Neutrosophic Logic 1), 55-137
PNL2 (Paraconsistent Neutrosophic Logic 2), 55-137
Predicate, 20-21
Principle of substitution, 89, 90, 91
Proposition, 7
Propositional calculus, 17

Q

Quantification theory, 19-22
Quantify, 19
Quantum mechanics, 29, 59

R

Reflexive property, 131, 133
Resolution, 18, 35, 47

S

Semigroup, 72, 73, 77
Shrodinger's cat, 29
Significant contradiction, 40
Significant tautology, 40
Singh, S., 52
Smarandache, Florentin, 3-4, 52-54, 138
Smarandache logic (see Neutrosophic logic), 54
Statistically significant, 40
Strong Kleene three-valued logic, 25-27

T

Tarski, A., 138
Tautology, 13-14, 36, 40, 93, 94
Temporal logic, 50-51
Theorem, 16

Transitive property, 18, 120, 121, 131, 137
Trichotomy property, 122

U
Unit resolution, 18, 35, 47
Universal quantifier, 20-22, 26, 28, 67, 68, 70, 71, 72
Universe of discourse, 20

V
Vacuous proof, 13
Valid, 35

W
Well-formed formulas (wffs) 15-16, 18-19, 33, 35, 47, 49, 94, 131, 132, 133

Z
Z-valid, 46
Zadeh, Lofti, 37, 57, 138

Neutrosophic Logic was created by Florentin Smarandache (1995) and is an extension / combination of the fuzzy logic, intuitionistic logic, paraconsistent logic, and the three-valued logics that use an indeterminate value.

Definition of Neutrosophic Logic:
Let T, I, F be standard or non-standard real subsets of the non-standard unit interval $]^-0, 1^+[$,
with sup T = t_sup, inf T = t_inf,
sup I = i_sup, inf I = i_inf,
sup F = f_sup, inf F = f_inf,
and n_sup = t_sup+i_sup+f_sup,
n_inf = t_inf+i_inf+f_inf.
Of course, $^-0 \leq$ n_inf $\leq$ n_sup $\leq 3^+$.
A logic in which each proposition is estimated to have the percentage of truth in a subset T, the percentage of indeterminacy in a subset I, and the percentage of falsity in a subset F, where T, I, F are defined above, is called *Neutrosophic Logic*.

The sets T, I, F are not necessarily intervals, but may be any real sub-unitary subsets: discrete or continuous; single-element, finite, or (countable or uncountable) infinite; union or intersection of various subsets; etc.
They may also overlap. The real subsets could represent the relative errors in determining t, i, f (in the case when the subsets T, I, F are reduced to points).
Statically T, I, F are subsets. But dynamically, looking therefore from another perspective, the components T, I, F are at each instance dependant on many parameters, and therefore they can be considered set-valued vector functions or even operators.
T, I, and F are called *neutrosophic components*, representing the truth, indeterminacy, and falsehood values respectively referring to neutrosophy, neutrosophic logic, neutrosophic set, neutrosophic probability, neutrosophic statistics.
This representation is closer to the reasoning of the human mind. It characterizes / catches the imprecision of knowledge or linguistic inexactitude perceived by various observers (that's why T, I, F are subsets - not necessarily single-elements), uncertainty due to incomplete knowledge or acquisition errors or stochasticity (that's why the subset I exists), and vagueness due to lack of clear contours or limits (that's why T, I, F are subsets and I exists; in particular for the appurtenance to the neutrosophic sets).

The advantage of using neutrosophic logic is that this logic distinguishes between relative truth, that is a truth in one or a few worlds only, noted by NL(relative truth)=1, and absolute truth, that is a truth in all possible worlds, noted by NL(absolute truth)=1^+. And similarly, neutrosophic logic distinguishes between relative falsehood, noted by 0, and absolute falsehood, noted by $^-0$.
In neutrosophic logic the sum of components is not necessarily 1 as in classical and fuzzy logic, but any number between $^-0$ and 3^+, and this allows the neutrosophic logic to be able to deal with paradoxes, propositions which are true and false in the same time: thus NL(paradox)=(1, I, 1); fuzzy logic can not do this because in fuzzy logic the sum of components has to be 1.

$ 19.95

ISBN: 1-931233-60-8

ISBN-13: 978-1-931233-60-6